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 №6:

«Проведення трьохфакторного експерименту при використанні рівняння регресії з квадратичними членами»

Виконав:

студент групи ІВ-81

Соколов В.С.

Залікова книжка № 8126

Перевірив:

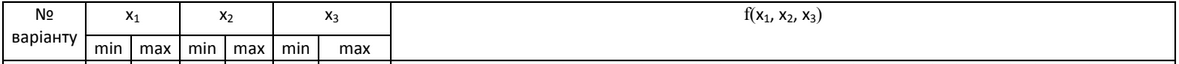
Регіда П. Г.

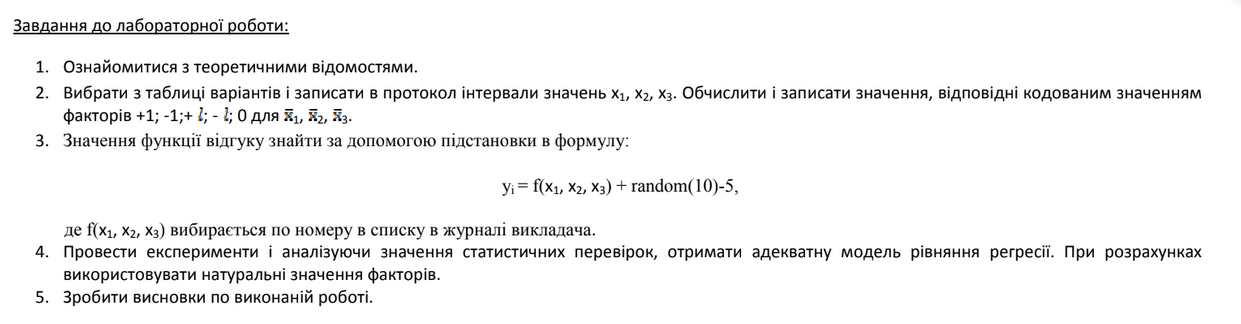
Київ 2020р.

**Мета:** Провести трьохфакторний експеримент і отримати адекватну модель – рівняння регресії, використовуючи рототабельний

композиційний план.

**Варіант:**

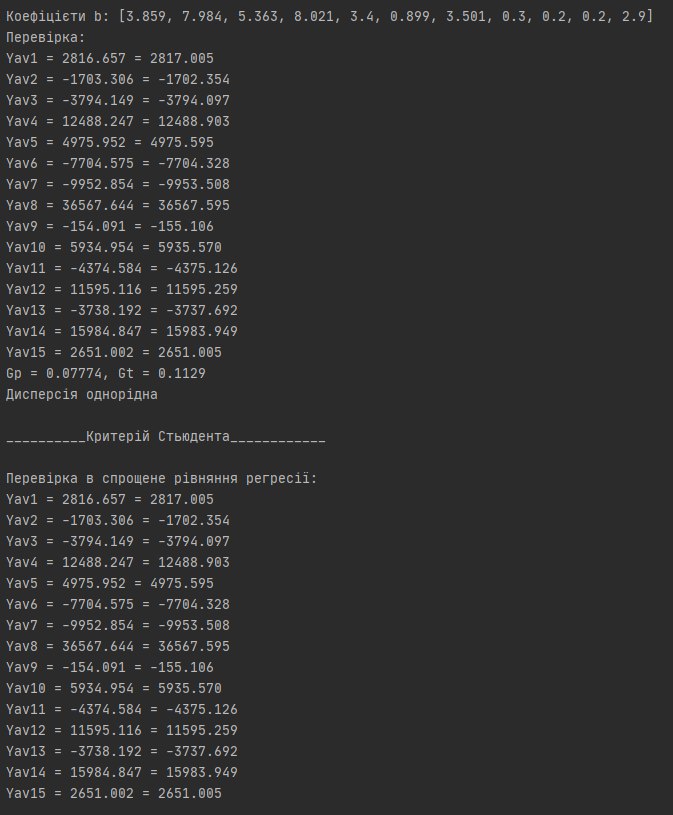
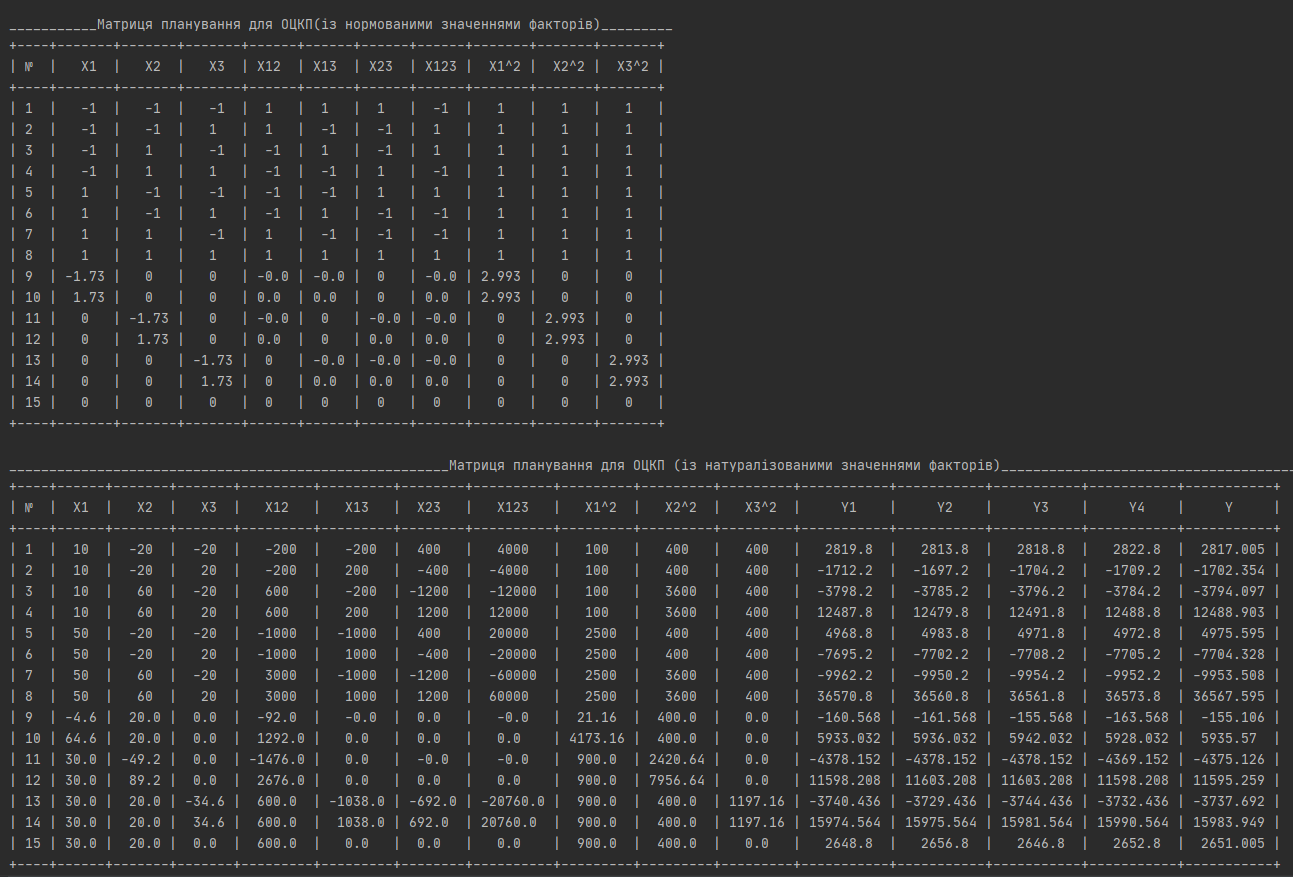


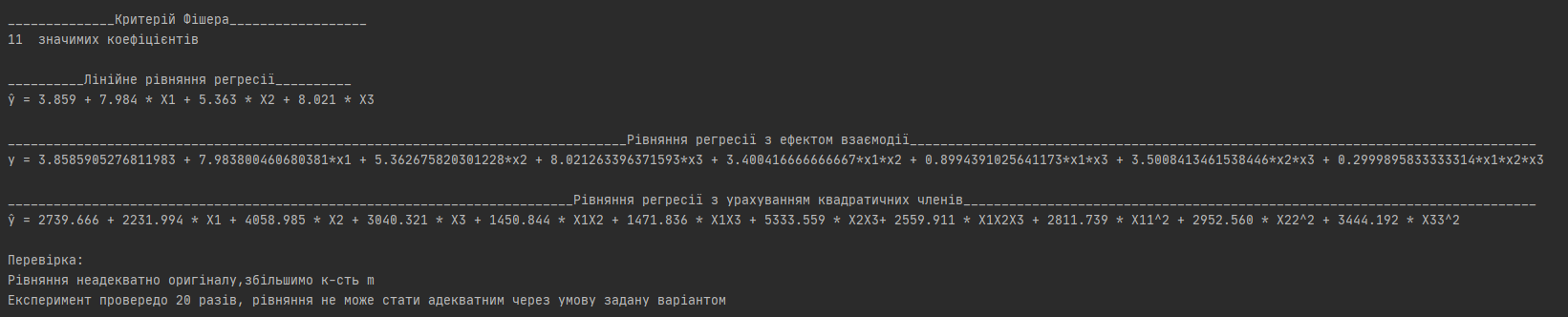


Лістинг програми:

import random  
from scipy.stats import f, t  
from prettytable import PrettyTable  
import numpy as np  
  
x1min = 10  
x1max = 50  
x2min = -20  
x2max = 60  
x3min = -20  
x3max = 20  
  
x01 = (x1max+x1min)/2  
x02 = (x2max+x2min)/2  
x03 = (x3max+x3min)/2  
deltax1 = x1max-x01  
deltax2 = x2max-x02  
deltax3 = x3max-x03  
  
m = 20  
  
X11 = [-1, -1, -1, -1, 1, 1, 1, 1, -1.73, 1.73, 0, 0, 0, 0, 0]  
X22 = [-1, -1, 1, 1, -1, -1, 1, 1, 0, 0, -1.73, 1.73, 0, 0, 0]  
X33 = [-1, 1, -1, 1, -1, 1, -1, 1, 0, 0, 0, 0, -1.73, 1.73, 0]  
  
  
def sumkf2(x1, x2):  
 return [round(x1[i] \* x2[i], 3) for i in range(len(x1))]  
  
  
def sumkf3(x1, x2, x3):  
 return [round(x1[i] \* x2[i] \* x3[i], 3) for i in range(len(x1))]  
  
  
def kv(x):  
 return [round(x[i]\*\*2, 3) for i in range(len(x))]  
  
  
flag = False  
counter = 0  
while True:  
 try:  
 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)  
  
 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)  
  
 print("\nf(x1,x2,x3) = 8.8 + 8.0\*X1 +5.4\*X2 + 8.0\*X3 + 0.2\*X1kv + 0.2\*X2kv + 2.9\*X3kv+ 3.4\*X12 + 0.9\*X13 +"  
 " 3,5\*X23 + 0.3\*X123")  
 print("\n\_\_\_\_\_\_\_\_\_\_\_Матриця планування для ОЦКП(із нормованими значеннями факторів)\_\_\_\_\_\_\_\_\_")  
 print(table1)  
  
 X1 = [x1min, x1min, x1min, x1min, x1max, x1max, x1max, x1max, round(-1.73\*deltax1+x01, 3),  
 round(1.73\*deltax1+x01, 3), x01, x01, x01, x01, x01]  
 X2 = [x2min, x2min, x2max, x2max, x2min, x2min, x2max, x2max, x02, x02, round(-1.73\*deltax2+x02, 3),  
 round(1.73\*deltax2+x02, 3), x02, x02, x02]  
 X3 = [x3min, x3max, x3min, x3max, x3min, x3max, x3min, x3max, x03, x03, x03, x03, round(-1.73\*deltax3+x03, 3),  
 round(1.73\*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)  
  
 # Y в діапазоні від 1 до 20, для 20 повторів збільшення m  
 Y = [[8.8 + 8.0\*X1[i] + 5.4\*X2[i] + 8.0\*X3[i] + 0.2\*X1kv[i] + 0.2\*X2kv[i] + 2.9\*X3kv[i] + 3.4\*X12[i] +  
 0.9\*X13[i] + 3.5\*X23[i] + 0.3\*X123[i] + random.randrange(-10, 10) - 5 for i in range(15)] for \_ in range(m)]  
  
 Yav = [sum([Y[i][k]/m for i in range(m)]) for k in range(15)]  
  
 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", X1kv)  
 table2.add\_column("X2^2", X2kv)  
 table2.add\_column("X3^2", X3kv)  
 for i in range(4):  
 table2.add\_column(f"Y{i+1}", [round(element, 3) for element in Y[i]])  
 table2.add\_column("Y", [round(element, 3) for element in Yav])  
  
 print("\n{:\_^180}".format("Матриця планування для ОЦКП (із натуралізованими значеннями факторів)"))  
 print(table2)  
  
 d = [sum([(Y[k][i] - Yav[i])\*\*2/m for k in range(m)]) for i in range(15)]  
  
 X0 = [1 for \_ in range(15)]  
 b = np.linalg.lstsq(list(zip(X0 , X1, X2, X3, X12, X13, X23, X123, X1kv, X2kv, X3kv)), Yav, rcond=None)[0]  
 br = [round(i, 3) for i in b]  
 print("\nКоефіцієти b:", br)  
 print("Перевірка:")  
 for i in range(15):  
 result = 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]  
 print(f"Yav{i+1} = {result:.3f} = {Yav[i]:.3f}")  
  
 Gp = max(d) / sum(d)  
 q = 0.05  
 f1 = m - 1  
 f2 = N = 15  
 fisher = f.isf(\*[q / f2, f1, (f2 - 1) \* f1])  
 Gt = fisher / (fisher + (f2 - 1))  
 print(f"Gp = {Gp:.4}, Gt = {Gt:.4}")  
  
 if Gp < Gt:  
 print("Дисперсія однорідна")  
 print("\n\_\_\_\_\_\_\_\_\_\_Критерій Стьюдента\_\_\_\_\_\_\_\_\_\_\_\_")  
 sb = sum(d) / N  
 ssbs = sb / N \* m  
 sbs = ssbs \*\* 0.5  
  
 beta = [0 for \_ in range(11)]  
 beta[0] = (Yav[0]\*1 + Yav[1]\*1 + Yav[2]\*1 + Yav[3]\*1 + Yav[4]\*1 + Yav[5]\*1 + Yav[6]\*1 + Yav[7]\*1 +  
 Yav[8]\*(-1.215) + Yav[9]\*1.215 + Yav[10]\*0 + Yav[11]\*0 + Yav[12]\*0 + Yav[13]\*0 + Yav[14]\*0) / 15  
 beta[1] = (Yav[0]\*(-1) + Yav[1]\*(-1) + Yav[2]\*(-1) + Yav[3]\*(-1) + Yav[4]\*1 + Yav[5]\*1 + Yav[6]\*1 +  
 Yav[7]\*1 + Yav[8]\*0 + Yav[9]\*0 + Yav[10]\*(-1.215) + Yav[11]\*1.215 + Yav[12] \*0 + Yav[13]\*0  
 + Yav[14]\*0) / 15  
 beta[2] = (Yav[0]\*(-1) + Yav[1]\*(-1) + Yav[2]\*1 + Yav[3]\*1 + Yav[4]\*(-1) + Yav[5]\*(-1) + Yav[6]\*1 + Yav[7]\*1  
 + Yav[8]\*0 + Yav[9]\*0 + Yav[10]\*0 + Yav[11]\*0 +  
 Yav[12]\*(-1.215) + Yav[13]\*1.215 + Yav[14]\*0) / 15  
 beta[3] = (Yav[0]\*(-1) + Yav[1]\*1 + Yav[2]\*(-1) + Yav[3]\*1 + Yav[4]\*(-1) +  
 Yav[5]\*1 + Yav[6]\*(-1) + Yav[7]\*1) / 15  
 beta[4] = (Yav[0]\*1 + Yav[1]\*1 + Yav[2]\*(-1) + Yav[3]\*(-1) + Yav[4]\*(-1) +  
 Yav[5]\*(-1) + Yav[6]\*1 + Yav[7]\*1) / 15  
 beta[5] = (Yav[0]\*1 + Yav[1]\*(-1) + Yav[2]\*1 + Yav[3]\*(-1) + Yav[4]\*(-1) +  
 Yav[5]\*1 + Yav[6]\*(-1) + Yav[7]\*1) / 15  
 beta[6] = (Yav[0]\*1 + Yav[1]\*(-1) + Yav[2]\*(-1) + Yav[3]\*1 + Yav[4]\*1 +  
 Yav[5]\*(-1) + Yav[6]\*(-1) + Yav[7]\*1) / 15  
 beta[7] = (Yav[0]\*(-1) + Yav[1]\*1 + Yav[2]\*1 + Yav[3]\*(-1) + Yav[4]\*1 +  
 Yav[5]\*(-1) + Yav[6]\*(-1) + Yav[7]\*1) / 15  
 beta[8] = (Yav[0]\*1 + Yav[1]\*1 + Yav[2]\*1 + Yav[3]\*1 + Yav[4]\*1 + Yav[5]\*1 + Yav[6]\*1 + Yav[7]\*1 +  
 Yav[8]\*1.46723 + Yav[9]\*1.46723) / 15  
 beta[9] = (Yav[0]\*1 + Yav[1]\*1 + Yav[2]\*1 + Yav[3]\*1 + Yav[4]\*1 + Yav[5]\*1 + Yav[6]\*1 + Yav[7]\*1 +  
 Yav[10]\*1.46723 + Yav[11]\*1.46723) / 15  
 beta[10] = (Yav[0]\*1 + Yav[1]\*1 + Yav[2]\*1 + Yav[3]\*1 + Yav[4]\*1 + Yav[5]\*1 + Yav[6]\*1 + Yav[7]\*1 +  
 Yav[12]\*1.46723 + Yav[13]\*1.46723) / 15  
  
 f3 = f1 \* f2  
 ttabl = abs(t.ppf(q / 2, f3))  
  
 d\_ = 11  
 for i in range(11):  
 if abs(beta[i] / sbs) < ttabl:  
 print("t%s <ttabl, b%s не значимий" % (i, i))  
 b[i] = 0  
 d\_ -= 1  
  
 print("\nПеревірка в спрощене рівняння регресії:")  
 for i in range(15):  
 result = 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]  
 print(f"Yav{i+1} = {result:.3f} = {Yav[i]:.3f}")  
  
 yy = [0 for \_ in range(15)]  
 yy[0] = 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  
 yy[1] = 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  
 yy[2] = 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  
 yy[3] = 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  
 yy[4] = 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  
 yy[5] = 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  
 yy[6] = 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  
 yy[7] = 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  
 yy[8] = 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]  
 yy[9] = 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]  
 yy[10] = 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]  
 yy[11] = 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]  
 yy[12] = 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]  
 yy[13] = 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]  
 yy[14] = 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 = sum([(yy[i] - Yav[i])\*\*2 for i in range(15)])\*(m/(N-d\_))  
 Fp = sad / sb  
 Ft = abs(f.isf(q, f4, f3))  
  
 print("\n\_\_\_\_\_\_\_\_\_\_Лінійне рівняння регресії\_\_\_\_\_\_\_\_\_\_\n"  
 "ŷ = {:.3f} + {:.3f} \* X1 + {:.3f} \* X2 + {:.3f} \* X3".format(b[0], b[1], b[2], b[3]))  
 print("\n{:\_^200}".format("Рівняння регресії з ефектом взаємодії"))  
 print("y = {} + {}\*x1 + {}\*x2 + {}\*x3 + {}\*x1\*x2 + {}\*x1\*x3 + {}\*x2\*x3 + {}\*x1\*x2\*x3". format(\*b))  
 print("\n{:\_^200}".format("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 \n\nПеревірка:"  
 .format(\*beta))  
  
 if Fp > Ft:  
 print("Рівняння неадекватно оригіналу,збільшимо к-сть m")  
 m += 1  
 counter += 1  
 else:  
 print("Рівняння адекватно оригіналу")  
 flag = True  
  
 else:  
 print("Дисперсія неоднорідна(збільшемо кількість дослідів)")  
 m += 1  
 counter += 1  
 finally:  
 if counter == 20:  
 print('Експеримент провередо 20 разів, рівняння не може стати адекватним через умову задану варіантом')  
 break  
 if flag:  
 break

1. Результат виконання роботи програми:





**Висновок:** Отже, у ході виконання лабораторної роботи №6 ми провели трьохфакторний експеримент і отримати адекватну модель – рівняння регресії, використовуючи рототабельний композиційний план