**ЛАБОРАТОРНА РОБОТА № 2**

з дисципліни «Методи оптимізації та планування експерименту» на тему

”ПРОВЕДЕННЯ ДВОФАКТОРНОГО ЕКСПЕРИМЕНТУ З

ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ”

ВИКОНАВ:

студент ІІ курсу ФІОТ

групи ІВ-83

Кочерук Д.А.

Варіант: 311

ПЕРЕВІРИВ:

Регіда П. Г.

Київ – 2020

ymax = (30 - 14)\*10 = 16\*10 = 160

ymin = (20 - 14)\*10 = 6\*10 = 60

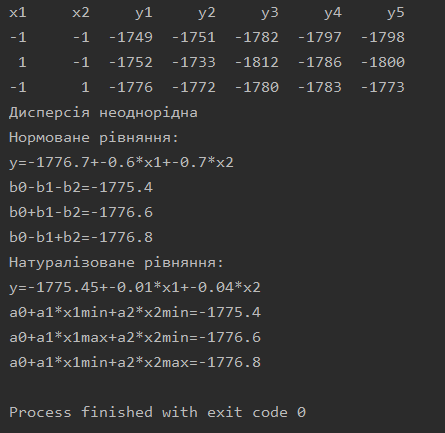




**Лістинг програми:**

from random import randint  
from math import sqrt  
  
  
def get\_r\_kr(m):  
 table\_values = {2: 1.73, 6: 2.16, 8: 2.43, 10: 2.62, 12: 2.75, 15: 2.9, 20: 3.08}  
 for i in range(len(table\_values.keys())):  
 if m == list(table\_values.keys())[i]:  
 return list(table\_values.values())[i]  
 if m > 20:  
 return list(table\_values.values())[-1]  
 if m > list(table\_values.keys())[i]:  
 less\_than\_m\_key = list(table\_values.keys())[i]  
 less\_than\_m = list(table\_values.values())[i]  
 more\_than\_m\_key = list(table\_values.keys())[i + 1]  
 more\_than\_m = list(table\_values.values())[i + 1]  
 return less\_than\_m + (more\_than\_m - less\_than\_m) \* (m - less\_than\_m\_key) / (  
 more\_than\_m\_key - less\_than\_m\_key)  
  
  
def determinant(matrix):  
 return matrix[0][0] \* matrix[1][1] \* matrix[2][2] + matrix[0][1] \* matrix[1][2] \* matrix[2][0] + matrix[0][2] \* \  
 matrix[1][0] \* matrix[2][1] - matrix[0][2] \* matrix[1][1] \* matrix[2][0] - matrix[0][1] \* matrix[1][0] \* \  
 matrix[2][2] - matrix[0][0] \* matrix[1][2] \* matrix[2][1]  
  
  
n\_variant = 10  
y\_min = (20 - n\_variant) \* 10  
y\_max = (30 - n\_variant) \* 10  
x1\_min = -25  
x1\_max = -5  
x2\_min = 10  
x2\_max = 60  
m = 5  
  
  
def main():  
 global m  
 response\_list1 = [randint(y\_min, y\_max) for i in range(m)]  
 response\_list2 = [randint(y\_min, y\_max) for j in range(m)]  
 response\_list3 = [randint(y\_min, y\_max) for k in range(m)]  
  
 average1 = sum(response\_list1) / len(response\_list1)  
 average2 = sum(response\_list2) / len(response\_list2)  
 average3 = sum(response\_list3) / len(response\_list3)  
  
 dispersion1 = sum((i - average1) \*\* 2 for i in response\_list1) / len(response\_list1)  
 dispersion2 = sum((i - average2) \*\* 2 for i in response\_list2) / len(response\_list2)  
 dispersion3 = sum((i - average3) \*\* 2 for i in response\_list3) / len(response\_list3)  
  
 major\_deviation = sqrt((4 \* m - 4) / (m \* m - 4 \* m))  
  
 f12 = dispersion1 / dispersion2 if dispersion1 >= dispersion2 else dispersion2 / dispersion1  
 f23 = dispersion2 / dispersion3 if dispersion2 >= dispersion3 else dispersion3 / dispersion2  
 f13 = dispersion1 / dispersion3 if dispersion1 >= dispersion3 else dispersion3 / dispersion1  
  
 t12 = (m - 2) / m \* f12  
 t23 = (m - 2) / m \* f23  
 t13 = (m - 2) / m \* f13  
  
 r12 = abs(t12 - 1) / major\_deviation  
 r23 = abs(t23 - 1) / major\_deviation  
 r13 = abs(t13 - 1) / major\_deviation  
  
 r\_kr = get\_r\_kr(m)  
  
 print(f'{y\_min=}')  
 print(f'{y\_max=}')  
  
 print(f'\nЗначення відгуку в діапазоні [{y\_min}-{y\_max}]:')  
 print(\*response\_list1, sep='\t')  
 print(\*response\_list2, sep='\t')  
 print(\*response\_list3, sep='\t')  
 print('\nСереднє значення відгуку в кожній з точок плану:')  
 print(average1)  
 print(average2)  
 print(average3)  
 print('\nДисперсії для кожної точки планування:')  
 print(f'{dispersion1:.3f}')  
 print(f'{dispersion2:.3f}')  
 print(f'{dispersion3:.3f}')  
 print('\nОсновне відхилення:')  
 print(f'{major\_deviation:.3f}')  
 print(f'\n{r12=:.3f}', '<' if r12 < r\_kr else '>', f'{r\_kr=:.3f}')  
 print(f'\n{r23=:.3f}', '<' if r23 < r\_kr else '>', f'{r\_kr=:.3f}')  
 print(f'\n{r13=:.3f}', '<' if r13 < r\_kr else '>', f'{r\_kr=:.3f}')  
  
 if r12 < r\_kr and r23 < r\_kr and r13 < r\_kr:  
 print('\nОднорідність підтверджується з ймовірністю 0.99\n')  
  
 normalized\_x1\_x2 = [  
 [-1, -1],  
 [-1, 1],  
 [1, -1]  
 ]  
  
 mx\_list = [sum(i) / len(i) for i in list(zip(normalized\_x1\_x2[0], normalized\_x1\_x2[1], normalized\_x1\_x2[2]))]  
 my = sum([average1, average2, average3]) / len([average1, average2, average3])  
 a1 = sum(i[0] \*\* 2 for i in normalized\_x1\_x2) / len(normalized\_x1\_x2)  
 a2 = sum(i[0] \* i[1] for i in normalized\_x1\_x2) / len(normalized\_x1\_x2)  
 a3 = sum(i[1] \*\* 2 for i in normalized\_x1\_x2) / len(normalized\_x1\_x2)  
 a11 = sum(  
 normalized\_x1\_x2[i][0] \* [average1, average2, average3][i] for i in range(len(normalized\_x1\_x2))) / len(  
 normalized\_x1\_x2)  
 a22 = sum(  
 normalized\_x1\_x2[i][1] \* [average1, average2, average3][i] for i in range(len(normalized\_x1\_x2))) / len(  
 normalized\_x1\_x2)  
 matrix\_b = [  
 [1, mx\_list[0], mx\_list[1]],  
 [mx\_list[0], a1, a2],  
 [mx\_list[1], a2, a3]  
 ]  
 matrix\_b1 = [  
 [my, mx\_list[0], mx\_list[1]],  
 [a11, a1, a2],  
 [a22, a2, a3]  
 ]  
 matrix\_b2 = [  
 [1, my, mx\_list[1]],  
 [mx\_list[0], a11, a2],  
 [mx\_list[1], a22, a3]  
 ]  
 matrix\_b3 = [  
 [1, mx\_list[0], my],  
 [mx\_list[0], a1, a11],  
 [mx\_list[1], a2, a22]  
 ]  
 b0 = determinant(matrix\_b1) / determinant(matrix\_b)  
 b1 = determinant(matrix\_b2) / determinant(matrix\_b)  
 b2 = determinant(matrix\_b3) / determinant(matrix\_b)  
  
 print('\nРозрахунок нормованих коефіцієнтів рівняння регресії:')  
  
 for i in normalized\_x1\_x2:  
 print(  
 f'y = b0 + b1 \* x1 + b2 \* x2 = {b0:.3f} + {b1:.3f} \* {i[0]:2} + {b2:.3f} \* {i[1]:2}'  
 f' = {b0 + b1 \* i[0] + b2 \* i[1]:.3f}')  
  
 x10 = (x1\_max + x1\_min) / 2  
 x20 = (x2\_max + x2\_min) / 2  
 delta\_x1 = (x1\_max - x1\_min) / 2  
 delta\_x2 = (x2\_max - x2\_min) / 2  
  
 a\_0 = b0 - b1 \* (x10 / delta\_x1) - b2 \* (x20 / delta\_x2)  
 a\_1 = b1 / delta\_x1  
 a\_2 = b2 / delta\_x2  
  
 print('\nЗапишемо натуралізоване рівняння регресії:')  
 print(  
 f'y = a0 + a1 \* x1 + a2 \* x2 = {a\_0:.3f} + {a\_1:.3f} \* {x1\_min:3} + {a\_2:.3f} \* {x2\_min:3}'  
 f' = {a\_0 + a\_1 \* x1\_min + a\_2 \* x2\_min:.3f}')  
 print(  
 f'y = a0 + a1 \* x1 + a2 \* x2 = {a\_0:.3f} + {a\_1:.3f} \* {x1\_min:3} + {a\_2:.3f} \* {x2\_max:3}'  
 f' = {a\_0 + a\_1 \* x1\_min + a\_2 \* x2\_max:.3f}')  
 print(  
 f'y = a0 + a1 \* x1 + a2 \* x2 = {a\_0:.3f} + {a\_1:.3f} \* {x1\_max:3} + {a\_2:.3f} \* {x2\_min:3}'  
 f' = {a\_0 + a\_1 \* x1\_max + a\_2 \* x2\_min:.3f}')  
  
 else:  
 print('\nОднорідність не підтвердилася, підвищуємо m на 1\n')  
 m += 1  
 main()  
  
  
main()

**Результат виконання роботи:**



**Контрольні запитання:**

1. Що таке регресійні поліноми і де вони застосовуються?

Регресійний поліном – це рівняння регресії виду



використовується в ТПЕ для оцінки результатів вимірів.

1. Визначення однорідності дисперсії.

Однорідність дисперсій – властивість, коли дисперсії вимірювання функцій відгуку є однаковими, або близькими.

1. Що називається повним факторним експериментом?

ПФЕ – експеримент, в якому використовуються всі можливі комбінації рівнів факторів.