Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Методи наукових досліджень

Лабораторна робота №3

**«ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ»**

Виконав:

Студент групи ІВ-91

Хандельди О.Р.

Варіант 126

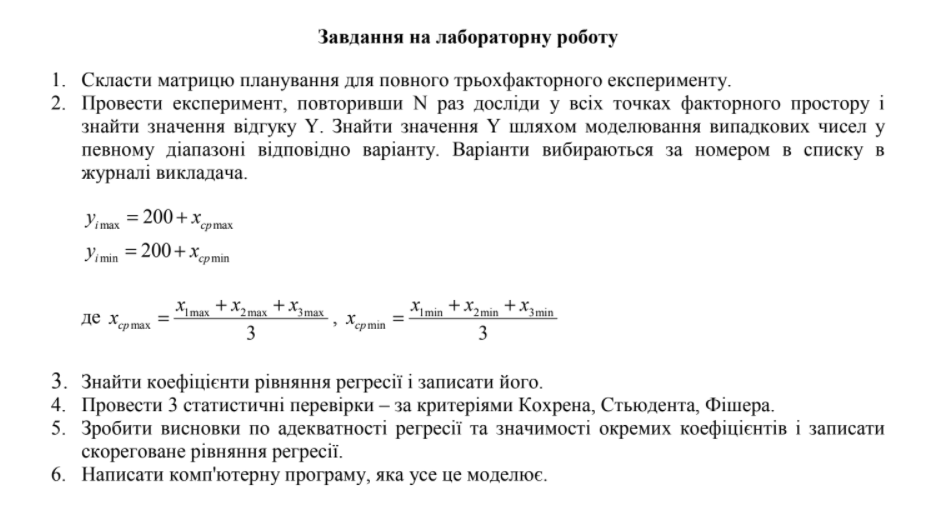
Перевірив:

Ас. Регіда П.Г.

Київ 2021 р.

**Мета:** Провести повний трьохфакторний експеримент. Знайти рівняння регресії адекватне об’єкту.

**Завдання на лабораторну роботу:**



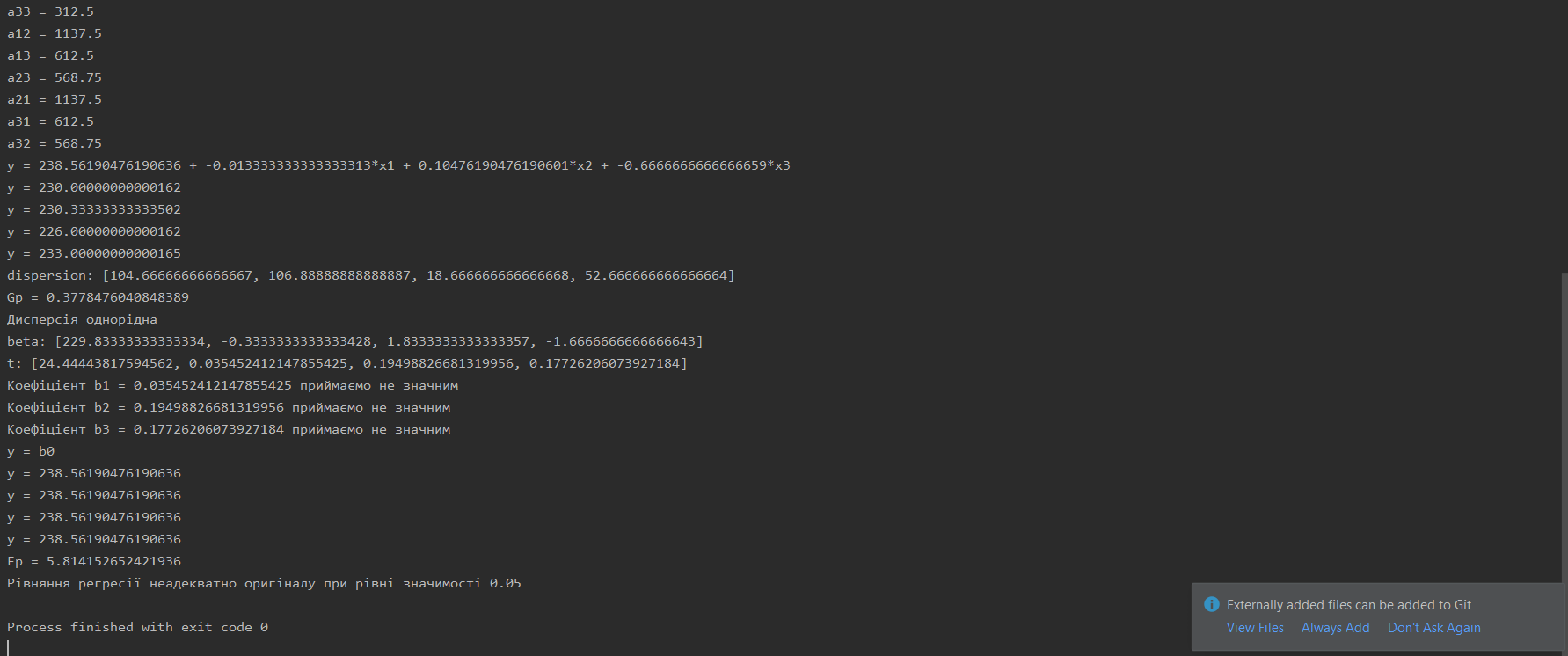
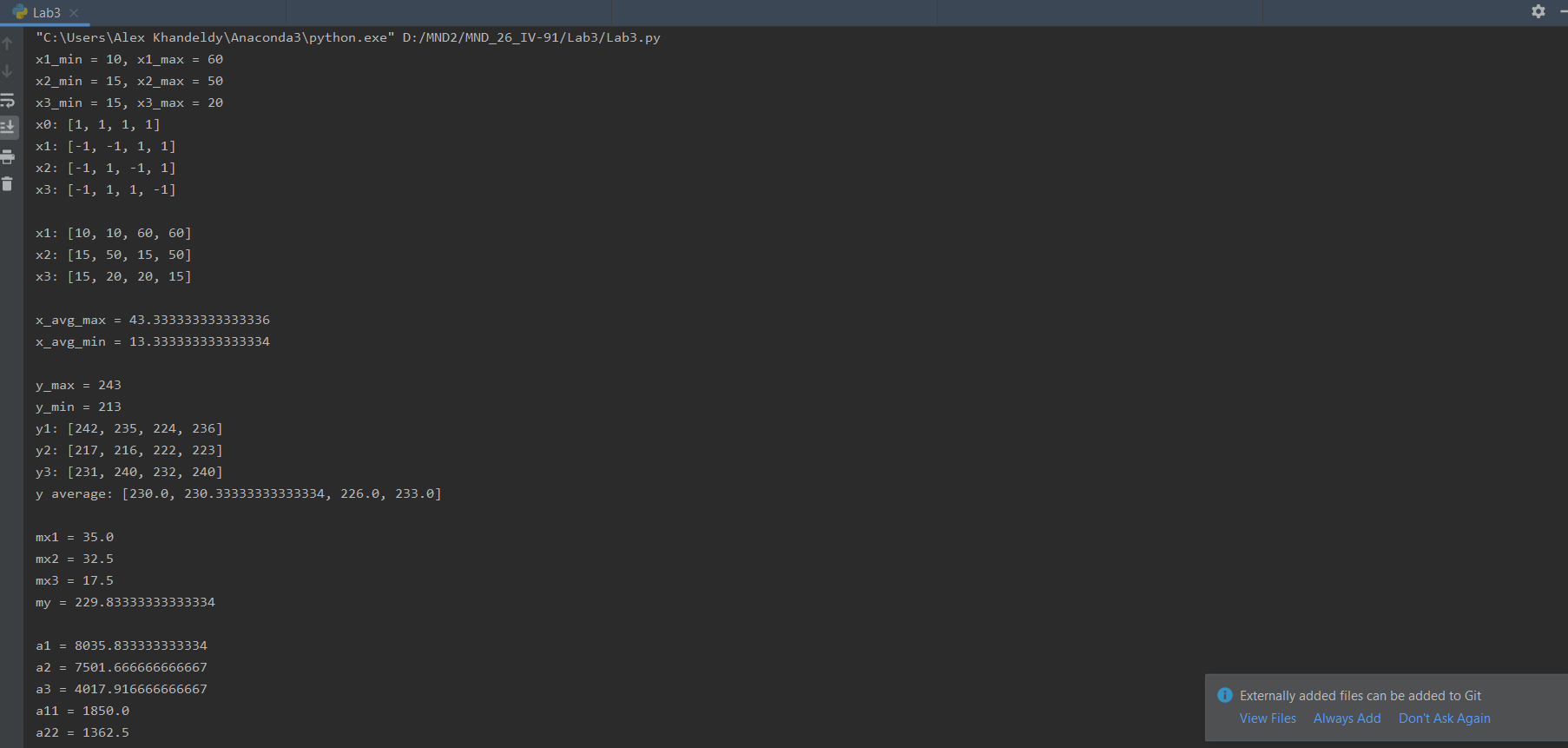
**Лістинг програми:**

from random import randint  
from numpy.linalg import det  
from functools import reduce  
  
  
def Naturalize(MatrixOfPlan, MinMaxArr):  
 result = []  
 for i in MatrixOfPlan:  
 result.append(MinMaxArr[1]) if i == 1 else result.append(MinMaxArr[0])  
 return result  
  
  
def main(m):  
 x1 = [10, 60]  
 x2 = [15, 50]  
 x3 = [15, 20]  
 print(f'x1\_min = {x1[0]}, x1\_max = {x1[1]}')  
 print(f'x2\_min = {x2[0]}, x2\_max = {x2[1]}')  
 print(f'x3\_min = {x3[0]}, x3\_max = {x3[1]}')  
  
 plan1x0 = [1, 1, 1, 1]  
 plan1x1 = [-1, -1, 1, 1]  
 plan1x2 = [-1, 1, -1, 1]  
 plan1x3 = [-1 \* (plan1x1[i] \* plan1x2[i]) for i in range(len(plan1x1))]  
 print('x0:', plan1x0)  
 print('x1:', plan1x1)  
 print('x2:', plan1x2)  
 print('x3:', plan1x3)  
  
 plan2x1 = Naturalize(plan1x1, x1)  
 plan2x2 = Naturalize(plan1x2, x2)  
 plan2x3 = Naturalize(plan1x3, x3)  
 print()  
 print('x1:', plan2x1)  
 print('x2:', plan2x2)  
 print('x3:', plan2x3)  
  
 x\_avg\_max = (max(plan2x1) + max(plan2x2) + max(plan2x3)) / 3  
 x\_avg\_min = (min(plan2x1) + min(plan2x2) + min(plan2x3)) / 3  
 print()  
 print(f'x\_avg\_max = {x\_avg\_max}')  
 print(f'x\_avg\_min = {x\_avg\_min}')  
  
 y\_max = int(200 + x\_avg\_max)  
 y\_min = int(200 + x\_avg\_min)  
 print()  
 print(f'y\_max = {y\_max}')  
 print(f'y\_min = {y\_min}')  
  
 y1 = [randint(y\_min, y\_max) for \_ in range(4)]  
 y2 = [randint(y\_min, y\_max) for \_ in range(4)]  
 y3 = [randint(y\_min, y\_max) for \_ in range(4)]  
 print('y1:', y1)  
 print('y2:', y2)  
 print('y3:', y3)  
  
 y\_avg\_arr = [(y1[i] + y2[i] + y3[i]) / 3 for i in range(4)]  
 print('y average:', y\_avg\_arr)  
  
 mx1 = reduce(lambda a, b: a + b, plan2x1) / 4  
 mx2 = reduce(lambda a, b: a + b, plan2x2) / 4  
 mx3 = reduce(lambda a, b: a + b, plan2x3) / 4  
 my = reduce(lambda a, b: a + b, y\_avg\_arr) / 4  
 print()  
 print(f'mx1 = {mx1}')  
 print(f'mx2 = {mx2}')  
 print(f'mx3 = {mx3}')  
 print(f'my = {my}')  
  
 a1 = sum([plan2x1[i] \* y\_avg\_arr[i] for i in range(4)]) / 4  
 a2 = sum([plan2x2[i] \* y\_avg\_arr[i] for i in range(4)]) / 4  
 a3 = sum([plan2x3[i] \* y\_avg\_arr[i] for i in range(4)]) / 4  
 print()  
 print(f'a1 = {a1}')  
 print(f'a2 = {a2}')  
 print(f'a3 = {a3}')  
  
 a11 = sum([i \* i for i in plan2x1]) / 4  
 a22 = sum([i \* i for i in plan2x2]) / 4  
 a33 = sum([i \* i for i in plan2x3]) / 4  
 print(f'a11 = {a11}')  
 print(f'a22 = {a22}')  
 print(f'a33 = {a33}')  
  
 a12 = sum([plan2x1[i] \* plan2x2[i] for i in range(4)]) / 4  
 a13 = sum([plan2x1[i] \* plan2x3[i] for i in range(4)]) / 4  
 a23 = sum([plan2x2[i] \* plan2x3[i] for i in range(4)]) / 4  
 a21 = a12  
 a31 = a13  
 a32 = a23  
 print(f'a12 = {a12}')  
 print(f'a13 = {a13}')  
 print(f'a23 = {a23}')  
 print(f'a21 = {a21}')  
 print(f'a31 = {a31}')  
 print(f'a32 = {a32}')  
  
 b0 = det([[my, mx1, mx2, mx3],  
 [a1, a11, a12, a13],  
 [a2, a21, a22, a23],  
 [a3, a31, a32, a33]]) / det([[1, mx1, mx2, mx3],  
 [mx1, a11, a12, a13],  
 [mx2, a21, a22, a23],  
 [mx3, a31, a32, a33]])  
 b1 = det([[1, my, mx2, mx3],  
 [mx1, a1, a12, a13],  
 [mx2, a2, a22, a23],  
 [mx3, a3, a32, a33]]) / det([[1, mx1, mx2, mx3],  
 [mx1, a11, a12, a13],  
 [mx2, a21, a22, a23],  
 [mx3, a31, a32, a33]])  
 b2 = det([[1, mx1, my, mx3],  
 [mx1, a11, a1, a13],  
 [mx2, a21, a2, a23],  
 [mx3, a31, a3, a33]]) / det([[1, mx1, mx2, mx3],  
 [mx1, a11, a12, a13],  
 [mx2, a21, a22, a23],  
 [mx3, a31, a32, a33]])  
 b3 = det([[1, mx1, mx2, my],  
 [mx1, a11, a12, a1],  
 [mx2, a21, a22, a2],  
 [mx3, a31, a32, a3]]) / det([[1, mx1, mx2, mx3],  
 [mx1, a11, a12, a13],  
 [mx2, a21, a22, a23],  
 [mx3, a31, a32, a33]])  
  
 print(f'y = {b0} + {b1}\*x1 + {b2}\*x2 + {b3}\*x3')  
  
 for i in range(4):  
 y = b0 + b1 \* plan2x1[i] + b2 \* plan2x2[i] + b3 \* plan2x3[i]  
 print('y =', y)  
  
 dispersion = [((y1[i] - y\_avg\_arr[i]) \*\* 2 + (y2[i] - y\_avg\_arr[i]) \*\* 2 + (y3[i] - y\_avg\_arr[i]) \*\* 2) / 3 for i in range(4)]  
 print('dispersion:', dispersion)  
  
 gp = max(dispersion) / sum(dispersion)  
 print('Gp =', gp)  
  
 # Рівень значимості q = 0.05; f1 = m - 1 = 2; f2 = N = 4  
 # За таблицею Gт = 0.7679  
 if gp < 0.7679:  
 print('Дисперсія однорідна')  
 else:  
 print('Дисперсія неоднорідна')  
  
 s2b = sum(dispersion) / 4  
 s2bs\_avg = s2b/4\*m  
 sb = s2bs\_avg \*\* (1/2)  
  
 beta0 = sum([y\_avg\_arr[i] \* plan1x0[i] for i in range(4)]) / 4  
 beta1 = sum([y\_avg\_arr[i] \* plan1x1[i] for i in range(4)]) / 4  
 beta2 = sum([y\_avg\_arr[i] \* plan1x2[i] for i in range(4)]) / 4  
 beta3 = sum([y\_avg\_arr[i] \* plan1x3[i] for i in range(4)]) / 4  
  
 beta\_arr = [beta0, beta1, beta2, beta3]  
 print('beta:', beta\_arr)  
 t\_arr = [abs(beta\_arr[i])/sb for i in range(4)]  
 print('t:', t\_arr)  
  
 # f3 = f1\*f2 = 2\*4 = 8  
 # З таблиці беремо значення 2.306  
 indexes = []  
 for i, v in enumerate(t\_arr):  
 if t\_arr[i] > 2.306:  
 indexes.append(i)  
 else:  
 print(f'Коефіцієнт b{i} = {v} приймаємо не значним')  
  
 b\_list = [b0, b1, b2, b3]  
 print(f'y = b{indexes[0]}')  
  
 b\_res = [b\_list[indexes[0]] for \_ in range(4)]  
 for i in b\_res:  
 print(f'y = {i}')  
  
 d = 1  
 s2\_ad = m \* sum([(y\_avg\_arr[i] - b\_res[i])\*\*2 for i in range(4)]) / 4 - d  
 fp = s2\_ad/s2b  
 print(f'Fp = {fp}')  
  
 # Fт = 4.5  
 if fp > 4.5:  
 print('Рівняння регресії неадекватно оригіналу при рівні значимості 0.05')  
 else:  
 print('Рівняння регресії адекватно оригіналу при рівні значимості 0.05')  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 main(m=5)

**Відповідь на контрольні питання:**

1. Дробовий факторний експеримент – частина ПФЕ, який мінімізує число дослідів, за рахунок тієї інформації, яка не дуже істотна для побудови моделі.
2. Значення Кохрена використовують для перевірки однорідності дисперсії.
3. Критерій Стьюдента перевіряє значущість коефіцієнтів рівняння.
4. Критерій Фішера використовують при перевірці отриманного рівняння регресії досліджуваному об’єкту.

**Результат виконання програми:**

****