Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

# ЛАБОРАТОРНА РОБОТА № 4

з дисципліни «Методи планування експерименту»

на тему «Проведення трьохфакторного експерименту при використанні рівняння регресії з урахуванням ефекту взаємодії..»

Виконав:
студент 2 курсу
групи ІВ-92
Дмитришин А. Д.
Номер у списку групи: 6

ПЕРЕВІРИВ:
ас. Регіда П.Г.

Київ - 2021

## Хід роботи

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

**Завдання:**

1. Скласти матрицю планування для повного трьохфакторного експерименту.
2. Провести експеримент, повторивши N раз досліди у всіх точках факторного простору і знайти значення відгуку Y. Знайти значення Y шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту. Варіанти вибираються за номером в списку в журналі викладача.

$$y_{i\max} = 200 + x_{cp\max}$$
$$y_{i\min} = 200 + x_{cp\min}$$

де $x_{cp\max} = \frac{x_{1\max} + x_{2\max} + x_{3\max}}{3}$, $x_{cp\min} = \frac{x_{1\min} + x_{2\min} + x_{3\min}}{3}$

| №варіанта | $x_1$ | | $x_2$ | | $x_3$ | |
|---|---|---|---|---|---|---|
| | min | max | min | Max | min | max |
| 206 | -30 | 20 | 25 | 45 | 25 | 30 |

3. Знайти коефіцієнти рівняння регресії і записати його.
4. Провести 3 статистичні перевірки – за критеріями Кохрена, Стьюдента, Фішера.
5. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.
6. Написати комп'ютерну програму, яка усе це моделює.

## Лістинг

```
package com.company;

import java.util.Random;

public class Lab4 {
    static double avarege(double[] y) {
        double sum = 0;
        for (int i = 0; i < y.length; i++) {
            sum += y[i];
        }
        return sum / y.length;
    }

    static double[] dispersion(double y[][]) {
        double d[] = new double[y.length];
        for (int i = 0; i < y.length; i++) {
            double sum = 0;
            for (int j = 0; j < y[i].length; j++) {
                sum += Math.pow(y[i][j] - avarege(y[i]), 2);
            }
            d[i] = sum / y[i].length;
        }
        return d;
    }

    static double max(double a[]) {
        double max = a[0];
        for (int i = 1; i < a.length; i++) {
            if (max < a[i]) max = a[i];
        }
        return max;
    }

    public static void main(String[] args) {
        double x[][] = {{-1, -1, 1, 1, -1, -1, 1, 1},
                {-1, 1, -1, 1, -1, 1, -1, 1},
                {-1, 1, 1, -1, 1, -1, -1, 1},
                {1, -1, -1, 1, 1, -1, -1, 1},
                {1, -1, 1, -1, -1, 1, -1, 1},
                {1, 1, -1, 1, -1, -1, 1, 1},
                {-1, -1, -1, -1, 1, 1, 1, 1}};
        double xm[][] = {{-30, -30, 20, 20, -30, -30, 20, 20},
                {25, 45, 25, 45, 25, 45, 25, 45},
                {25, 30, 30, 25, 30, 25, 25, 30},
                {-750, -1350, 500, 900, -750, -1350, 500, 900},
                {-750, -900, 600, 500, -900, -750, 500, 600},
                {625, 1350, 750, 1125, 750, 1125, 625, 1350},
                {-18750, -40500, 15000, 22500, -22500, -33750, 12500, 27000}};
        double xmin[] = {-30, 25, 25};
        double xmax[] = {20, 45, 30};
        double xmax_avr = avarege(xmax);
        double xmin_avr = avarege(xmin);
        double ymax = 200 + xmax_avr;
        double ymin = 200 + xmin_avr;
        int y1[] = new int[8];
        int y2[] = new int[8];
        int y3[] = new int[8];
        Random random = new Random();
        for (int i = 0; i < y1.length; i++) {
            y1[i] = random.nextInt((int) ymax - (int) ymin + 1) + (int) ymin;
            y2[i] = random.nextInt((int) ymax - (int) ymin + 1) + (int) ymin;
            y3[i] = random.nextInt((int) ymax - (int) ymin + 1) + (int) ymin;
```

```
        }
        double y[][] = {{y1[0], y2[0], y3[0]},
                {y1[1], y2[1], y3[1]},
                {y1[2], y2[2], y3[2]},
                {y1[3], y2[3], y3[3]},
                {y1[4], y2[4], y3[4]},
                {y1[5], y2[5], y3[5]},
                {y1[6], y2[6], y3[6]},
                {y1[7], y2[7], y3[7]}
        };
        double S[] = dispersion(y);
        double Gp = max(S) / (S[0] + S[1] + S[2] + S[3] + S[4] + S[5] + S[6] +
S[7]);
        double Gt = 0.5157;
        if (Gp > Gt) {
            System.out.println("Неоднорідна дисперсія");
            System.exit(1337);
        }
        double yavr[] = {avarege(y[0]), avarege(y[1]), avarege(y[2]), avarege(y[3]),
avarege(y[4]), avarege(y[5]), avarege(y[6]), avarege(y[7])};
        double b0 = (yavr[0] + yavr[1] + yavr[2] + yavr[3] + yavr[4] + yavr[5] +
yavr[6] + yavr[7]) / 8;
        double b1 = (yavr[0] * x[0][0] + yavr[1] * x[0][1] + yavr[2] * x[0][2] +
yavr[3] * x[0][3] + yavr[4] * x[0][4] +
                yavr[5] * x[0][5] + yavr[6] * x[0][6] + yavr[7] * x[0][7]) / 8;
        double b2 = (yavr[0] * x[1][0] + yavr[1] * x[1][1] + yavr[2] * x[1][2] +
yavr[3] * x[1][3] + yavr[4] * x[1][4] +
                yavr[5] * x[1][5] + yavr[6] * x[1][6] + yavr[7] * x[1][7]) / 8;
        double b3 = (yavr[0] * x[2][0] + yavr[1] * x[2][1] + yavr[2] * x[2][2] +
yavr[3] * x[2][3] + yavr[4] * x[2][4] +
                yavr[5] * x[2][5] + yavr[6] * x[2][6] + yavr[7] * x[2][7]) / 8;
        double b12 = (yavr[0] * x[0][0] * x[1][0] + yavr[1] * x[0][1] * x[1][1] +
yavr[2] * x[0][2] * x[1][2] +
                yavr[3] * x[0][3] * x[1][3] + yavr[4] * x[0][4] * x[1][4] + yavr[5]
* x[0][5] * x[1][5] +
                yavr[6] * x[0][6] * x[1][6] + yavr[7] * x[0][7] * x[1][7]) / 8;
        double b13 = (yavr[0] * x[0][0] * x[2][0] + yavr[1] * x[0][1] * x[2][1] +
yavr[2] * x[0][2] * x[2][2] +
                yavr[3] * x[0][3] * x[2][3] + yavr[4] * x[0][4] * x[2][4] + yavr[5]
* x[0][5] * x[2][5] +
                yavr[6] * x[0][6] * x[2][6] + yavr[7] * x[0][7] * x[2][7]) / 8;
        double b23 = (yavr[0] * x[1][0] * x[2][0] + yavr[1] * x[1][1] * x[2][1] +
yavr[2] * x[1][2] * x[2][2] +
                yavr[3] * x[1][3] * x[2][3] + yavr[4] * x[1][4] * x[2][4] + yavr[5]
* x[1][5] * x[2][5] +
                yavr[6] * x[1][6] * x[2][6] + yavr[7] * x[1][7] * x[2][7]) / 8;
        double b123 = (yavr[0] * x[0][0] * x[1][0] * x[2][0] + yavr[1] * x[0][1] *
x[1][1] * x[2][1] +
                yavr[2] * x[0][2] * x[1][2] * x[2][2] + yavr[3] * x[0][3] * x[1][3]
* x[2][3] +
                yavr[4] * x[0][4] * x[1][4] * x[2][4] + yavr[5] * x[0][5] * x[1][5]
* x[2][5] +
                yavr[6] * x[0][6] * x[1][6] * x[2][6] + yavr[7] * x[0][7] * x[1][7]
* x[2][7]) / 8;

        double ypr1 = b0 + b1 * x[0][0] + b2 * x[1][0] + b3 * x[2][0] + b12 *
x[3][0] + b13 * x[4][0] + b23 * x[5][0] + b123 * x[6][0];
        double ypr2 = b0 + b1 * x[0][1] + b2 * x[1][1] + b3 * x[2][1] + b12 *
x[3][1] + b13 * x[4][1] + b23 * x[5][1] + b123 * x[6][1];
        double ypr3 = b0 + b1 * x[0][2] + b2 * x[1][2] + b3 * x[2][2] + b12 *
x[3][2] + b13 * x[4][2] + b23 * x[5][2] + b123 * x[6][2];
        double ypr4 = b0 + b1 * x[0][3] + b2 * x[1][3] + b3 * x[2][3] + b12 *
x[3][3] + b13 * x[4][3] + b23 * x[5][3] + b123 * x[6][3];
        double ypr5 = b0 + b1 * x[0][4] + b2 * x[1][4] + b3 * x[2][4] + b12 *
```

```
x[3][4] + b13 * x[4][4] + b23 * x[5][4] + b123 * x[6][4];
        double ypr6 = b0 + b1 * x[0][5] + b2 * x[1][5] + b3 * x[2][5] + b12 *
x[3][5] + b13 * x[4][5] + b23 * x[5][5] + b123 * x[6][5];
        double ypr7 = b0 + b1 * x[0][6] + b2 * x[1][6] + b3 * x[2][6] + b12 *
x[3][6] + b13 * x[4][6] + b23 * x[5][6] + b123 * x[6][6];
        double ypr8 = b0 + b1 * x[0][7] + b2 * x[1][7] + b3 * x[2][7] + b12 *
x[3][7] + b13 * x[4][7] + b23 * x[5][7] + b123 * x[6][7];

        double Sbs = Math.sqrt((S[0] + S[1] + S[2] + S[3] + S[4] + S[5] + S[6] +
S[7]) / (8*8*3));
        double t[] = {Math.abs(b0) / Sbs, Math.abs(b1) / Sbs, Math.abs(b2) / Sbs,
Math.abs(b3) / Sbs,Math.abs(b12) / Sbs,
                Math.abs(b13) / Sbs,Math.abs(b23) / Sbs,Math.abs(b123) / Sbs,};
        int n = 0;
        String k = "";
        double bmat[] = {b0, b1, b2, b3,b12,b13,b23,b123};
        for (int i = 0; i < t.length; i++) {
            if (t[i] < 2.306) {
                k += "b" + i + " ";
                n += 1;
                bmat[i] = 0;
            }
        }
        double ys1 = bmat[0] + bmat[1] * x[0][0] + bmat[2] * x[1][0] + bmat[3] *
x[2][0] + bmat[4] * x[3][0] + bmat[5] * x[4][0] + bmat[6] * x[5][0] + bmat[7] *
x[6][0];
        double ys2 = bmat[0] + bmat[1] * x[0][1] + bmat[2] * x[1][1] + bmat[3] *
x[2][1] + bmat[4] * x[3][1] + bmat[5] * x[4][1] + bmat[6] * x[5][1] + bmat[7] *
x[6][1];
        double ys3 = bmat[0] + bmat[1] * x[0][2] + bmat[2] * x[1][2] + bmat[3] *
x[2][2] + bmat[4] * x[3][2] + bmat[5] * x[4][2] + bmat[6] * x[5][2] + bmat[7] *
x[6][2];
        double ys4 = bmat[0] + bmat[1] * x[0][3] + bmat[2] * x[1][3] + bmat[3] *
x[2][3] + bmat[4] * x[3][3] + bmat[5] * x[4][3] + bmat[6] * x[5][3] + bmat[7] *
x[6][3];
        double ys5 = bmat[0] + bmat[1] * x[0][4] + bmat[2] * x[1][4] + bmat[3] *
x[2][4] + bmat[4] * x[3][4] + bmat[5] * x[4][4] + bmat[6] * x[5][4] + bmat[7] *
x[6][4];
        double ys6 = bmat[0] + bmat[1] * x[0][5] + bmat[2] * x[1][5] + bmat[3] *
x[2][5] + bmat[4] * x[3][5] + bmat[5] * x[4][5] + bmat[6] * x[5][5] + bmat[7] *
x[6][5];
        double ys7 = bmat[0] + bmat[1] * x[0][6] + bmat[2] * x[1][6] + bmat[3] *
x[2][6] + bmat[4] * x[3][6] + bmat[5] * x[4][6] + bmat[6] * x[5][6] + bmat[7] *
x[6][6];
        double ys8 = bmat[0] + bmat[1] * x[0][7] + bmat[2] * x[1][7] + bmat[3] *
x[2][7] + bmat[4] * x[3][7] + bmat[5] * x[4][7] + bmat[6] * x[5][7] + bmat[7] *
x[6][7];
        int f4 = (8 - (8 - n));
        //f3 = (m-1)*N= 2*8=16
        double[] Fisher = {4.5, 3.6, 3.2, 3.0,};
        double Sad = 3 / (8 - (double)(8 - n)) * (Math.pow(yavr[0] - ys1, 2) +
Math.pow(yavr[1] - ys2, 2) +
                Math.pow(yavr[2] - ys3, 2) + Math.pow(yavr[3] - ys4,
2)+Math.pow(yavr[4] - ys5, 2) +
                Math.pow(yavr[5] - ys6, 2) + Math.pow(yavr[6] - ys7, 2) +
Math.pow(yavr[7] - ys8, 2));
        double Fp = Sad / Sbs;
        System.out.println("Нормована матриця:");
        for (int i = 0; i < x.length; i++) {
            for (int j = 0; j < x[i].length; j++) {
                System.out.print(x[i][j]+ " ");
            }
            System.out.println();
        }
```

```
        System.out.println("Середні значення: "+yavr[0] + " " + yavr[1] + " " +
yavr[2] + " " + yavr[3] + " " + yavr[4] + " " + yavr[5] + " " + yavr[6] + " " +
yavr[7]);
        System.out.println("Нормовані коефіцієнти: "+b0 + " " + b1 + " " + b2 + " "
+ b3 + " " + b12 + " " + b13 + " " + b23 + " " + b123);
        System.out.println("Значення з нормованими коефіцієнтами: "+ypr1 + " " +
ypr2 + " " + ypr3 + " " + ypr4 + " " + ypr5 + " " + ypr6 + " " + ypr7 + " " + ypr8);
        System.out.println(k + "коефіцієнти рівняння регресії приймаємо незначними
при рівні значимості 0.05, тобто вони виключаються з рівняння.");
        System.out.println("Значення отримані за критерієм СтЬюдента: "+ys1 + " " +
ys2 + " " + ys3 + " " + ys4+" " +ys5 + " " + ys6 + " " + ys7 + " " + ys8);
        if (Fp < Fisher[8 - n - 1]) {
            System.out.println("Fp=" + Fp + " Ft=" + Fisher[8 - n - 1] + " рівняння
регресії адекватно оригіналу при рівні значимості 0.05");
        } else
            System.out.println("Fp=" + Fp + " Ft=" + Fisher[8 - n - 1] + " рівняння
регресії неадекватно оригіналу при рівні значимості 0.05");

        }
}
```

## Результати роботи програми

```
Нормована матриця:
-1.0 -1.0 1.0 1.0 -1.0 -1.0 1.0 1.0
-1.0 1.0 -1.0 1.0 -1.0 1.0 -1.0 1.0
-1.0 1.0 1.0 -1.0 1.0 -1.0 -1.0 1.0
1.0 -1.0 -1.0 1.0 1.0 -1.0 -1.0 1.0
1.0 -1.0 1.0 -1.0 -1.0 1.0 -1.0 1.0
1.0 1.0 -1.0 1.0 -1.0 -1.0 1.0 1.0
-1.0 -1.0 -1.0 -1.0 1.0 1.0 1.0 1.0
Середні значення: 214.66666666666666 219.66666666666666 218.33333333333334 222.0 221.0 209.33333333333334 216.0 216.66666666666666
Нормовані коефіцієнти: 217.20833333333331 1.0416666666666679 -0.29166666666666785 1.7083333333333321 1.3749999999999964 -2.458333333333332 -0.45833333333333925 -1.4583333333333325
Значення з нормованими коефіцієнтами: 214.6666666666666 219.66666666666666 218.3333333333333 221.08333333333333 221.00000000000003 209.33333333333334 215.99999999999997 216.666666
b1 b2 b3 b4 b5 b6 b7 коефіцієнти рівняння регресії приймаємо незначними при рівні значимості 0.05, тобто вони виключаються з рівняння.
Значення отримані за критерієм СтЬюдента: 217.20833333333331 217.20833333333331 217.20833333333331 217.20833333333331 217.20833333333331 217.20833333333331 217.20833333333331 217
Fp=37.117935808720816 Ft=4.5 рівняння регресії неадекватно оригіналу при рівні значимості 0.05
```