Міністерство освіти і науки України

Національний технічний університет України «КПІ» імені Ігоря Сікорського

Кафедра обчислювальної техніки ФІОТ

ЗВІТ

з лабораторної роботи №4

з навчальної дисципліни «МЕТОДИ ОБЧИСЛЕННЯ ТА ПЛАНУВАННЯ ЕКСПЕРИМЕНТУ»

Тема:

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

Виконав:

студент 2-го курсу ФІОТ

групи ІО-91

Денисенко М. О.

Варіант - 7

Перевірив:

Регіда П. Г.

Київ 2021

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

Завдання:

1. Скласти матрицю планування для дробового 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.

$y_{max} = 200 + x_{cp\ max}$;

$y_{min} = 200 + x_{cp\ min}$,

де $x_{cp\ max} = (1/3)(x_{1max}+x_{2max}+x_{3max})$, $x_{cp\ min} = (1/3)(x_{1min}+x_{2min}+x_{3min})$

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 виконує.

107	10	40	25	45	40	45

Код програми:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

public class MopeLab4 {
   public static double determinant(double[][] arr) {
      double result = 0;
      if (arr.length == 1) {
         result = arr[0][0];
         return result;
      }
      else if (arr.length == 2) {
         result = arr[0][0] * arr[1][1] - arr[0][1] * arr[1][0];
         return result;
      }
      else {
         for (int i = 0; i < arr[0].length; i++) {
            double[][] temp = new double[arr.length - 1][arr[0].length - 1];

            for (int j = 1; j < arr.length; j++) {
               for (int k = 0; k < arr[0].length; k++) {
```

```
                if (k < i) temp[j - 1][k] = arr[j][k];
                else if (k > i) temp[j - 1][k - 1] = arr[j][k];
            }
        }
        result += arr[0][i] * Math.pow(-1, i) * determinant(temp);
    }
    return result;
  }
}

public static void main(String[] args) {


  int MinX1 = 10;
  int MaxX1 = 40;
  int MinX2 = 25;
  int MaxX2 = 45;
  int MinX3 = 40;
  int MaxX3 = 45;

  int m = 3;

  double MaxY = 243.333;
  double MinY = 225;

  int[][] x = {
        {1, -1, -1, -1},
        {1, -1, 1, 1},
        {1, 1, -1, 1},
        {1, 1, 1, -1}
  };

  int[][] xArr = {
        {MinX1, MinX2, MinX3},
        {MinX1, MaxX2, MaxX3},
        {MaxX1, MinX2, MaxX3},
        {MaxX1, MaxX2, MinX3}
  };

  double[][] aKoef = new double[3][3];
  List<String> symbols = new ArrayList<>();
  double[] mx = new double[3];
  double sum;
  double my;
  double[] a = new double[3];
  double[] yAverage = new double[4];
  double[] bArr = new double[4];
  double[] dispersionArr = new double[4];
  int f1 = 0;
  int f2 = 0;
  double q = 0;
  boolean work = false;
  boolean restart = false;

  Scanner st = new Scanner(System.in);
  System.out.println("Задайте значення m: ");
  try {
    m = st.nextInt();
    if (m > 0) {
      symbols.add("0");
      symbols.add("1");
      symbols.add("2");
      symbols.add("3");
```

```
            if (m > 3){
               for (int i = 3; i <= m; i++) {
                  symbols.add("" + i);
               }
            }
            symbols.add("\u2219"); // знак умножения (dot)
            symbols.add("\u2260"); // знак "не равно" (not_eq)
            work = true;
            restart = true;
         } else {
            System.out.println("Потрібно задати додатнє значення...");
         }
      } catch (Exception e) {
         System.out.println("Потрібно задати ціле значення...");
         m = 0;
      }

      while (restart) {
         while (work) {
            int dot = symbols.size()-2;
            int not_eq = symbols.size()-1;
            List<double[]> y = new ArrayList<>();
            System.out.printf("Лінійне рівняння регресії для нормованих значень x має вигляд : " +
                  "y = b%s + b%s%sx%s + b%s%sx%s + b%s%sx%s\n",
               symbols.get(0), symbols.get(1), symbols.get(dot), symbols.get(1),
               symbols.get(2), symbols.get(dot), symbols.get(2),
               symbols.get(3),symbols.get(dot),symbols.get(3));
            System.out.println();

            System.out.println("Нормована матриця планування експерименту : ");
            System.out.printf("X%s\tX%s\tX%s\tX%s\t", symbols.get(0), symbols.get(1), symbols.get(2),
symbols.get(3));
            for (int i = 1; i <= m; i++) {
               System.out.print("Y" + symbols.get(i) + "\t\t\t\t");
            }
            System.out.println();
            for (int i = 0; i < 4; i++) {
               double[] yTemp = new double[m];
               for (int j = 0; j < 4; j++) {
                  if (x[i][j] > 0){
                     System.out.print(" " + x[i][j] + "\t");
                  }
                  else System.out.print(x[i][j] + "\t");
               }
               for (int j = 0; j < m; j++) {
                  yTemp[j] = (Math.random() * (MaxY - MinY)) + MinY;
                  StringBuilder Y = new StringBuilder("" + (float) yTemp[j]);
                  /* Дабы колонки были ровными, и текст не съезжал, я добавил проверку на количество
символов в строке:
                  в конец каждого значения будут добавляться нули, чтоб ширина колонки была 9 символов*/
                  if (Y.length() < 9){
                     for (int k = Y.length(); k < 9; k++) {
                        Y.append("0");
                     }
                     System.out.print(Y + "\t\t");
                  }
                  else System.out.print(Y + "\t\t");
               }
               System.out.println();
               y.add(yTemp);
            }
            System.out.println();
```

```
System.out.println("Матриця планування експерименту : ");
System.out.printf("X%s\tX%s\tX%s\t", symbols.get(1), symbols.get(2), symbols.get(3));
for (int i = 1; i <= m; i++) {
   System.out.print("Y" + symbols.get(i) + "\t\t\t\t");
}
System.out.println();
for (int i = 0; i < 4; i++) {
   double[] yTemp;
   for (int j = 0; j < 3; j++) {
      System.out.print(xArr[i][j] + "\t");
   }
   yTemp = y.get(i);
   for (int j = 0; j < m; j++) {
      StringBuilder Y = new StringBuilder("" + (float) yTemp[j]);
      /* Дабы колонки были ровными, и текст не съезжал, я добавил проверку на количество
символов в строке:
      в конец каждого значения будут добавляться нули, чтоб ширина колонки была 9 символов*/
      if (Y.length() < 9){
         for (int k = Y.length(); k < 9; k++) {
            Y.append("0");
         }
         System.out.print(Y + "\t\t");
      }
      else System.out.print(Y + "\t\t");
   }
   System.out.println();
}

for (int i = 0; i < 4; i++) {
   sum = 0;
   double[] yTemp;
   yTemp = y.get(i);
   for (int j = 0; j < m; j++) {
      sum += yTemp[j];
   }
   yAverage[i] = sum / m;
}

for (int i = 0; i < 3; i++) {
   sum = 0;
   for (int j = 0; j < 4; j++) {
      sum += xArr[j][i];
   }
   mx[i] = sum / 4;
}
sum = 0;
for (int i = 0; i < 4; i++) {
   sum += yAverage[i];
}
my = sum / 4;

for (int i = 0; i < 3; i++) {
   sum = 0;
   for (int j = 0; j < 4; j++) {
      sum += xArr[j][i] * yAverage[j];
   }
   a[i] = sum / 4;
}

for (int i = 0; i < 3; i++) {
   sum = 0;
   for (int j = 0; j < 4; j++) {
      sum += Math.pow(xArr[j][i], 2);
```

```
            }
            aKoef[i][i] = sum / 4;
        }

        aKoef[0][1] = aKoef[1][0] = (xArr[0][0] * xArr[0][1] + xArr[1][0] * xArr[1][1] + xArr[2][0] * xArr[2][1]
+ xArr[3][0] * xArr[3][1]) / 4.;
        aKoef[0][2] = aKoef[2][0] = (xArr[0][0] * xArr[0][2] + xArr[1][0] * xArr[1][2] + xArr[2][0] * xArr[2][2]
+ xArr[3][0] * xArr[3][2]) / 4.;
        aKoef[1][2] = aKoef[2][1] = (xArr[0][1] * xArr[0][2] + xArr[1][1] * xArr[1][2] + xArr[2][1] * xArr[2][2]
+ xArr[3][1] * xArr[3][2]) / 4.;

        double[][] matrixTemp1 = {
                {my, mx[0], mx[1], mx[2]},
                {a[0], aKoef[0][0], aKoef[0][1], aKoef[0][2]},
                {a[1], aKoef[0][1], aKoef[1][1], aKoef[2][1]},
                {a[2], aKoef[0][2], aKoef[1][2], aKoef[2][2]}
        };

        double[][] matrixTemp2 = {
                {1, mx[0], mx[1], mx[2]},
                {mx[0], aKoef[0][0], aKoef[0][1], aKoef[0][2]},
                {mx[1], aKoef[0][1], aKoef[1][1], aKoef[2][1]},
                {mx[2], aKoef[0][2], aKoef[1][2], aKoef[2][2]}
        };

        bArr[0] = determinant(matrixTemp1) / determinant(matrixTemp2);

        double[][] matrixTemp3 = {
                {1, my, mx[1], mx[2]},
                {mx[0], a[0], aKoef[0][1], aKoef[0][2]},
                {mx[1], a[1], aKoef[1][1], aKoef[2][1]},
                {mx[2], a[2], aKoef[1][2], aKoef[2][2]}
        };

        bArr[1] = determinant(matrixTemp3) / determinant(matrixTemp2);

        double[][] matrixTemp4 = {
                {1, mx[0], my, mx[2]},
                {mx[0], aKoef[0][0], a[0], aKoef[0][2]},
                {mx[1], aKoef[0][1], a[1], aKoef[2][1]},
                {mx[2], aKoef[0][2], a[2], aKoef[2][2]}
        };
        bArr[2] = determinant(matrixTemp4) / determinant(matrixTemp2);

        double[][] matrixTemp5 = {
                {1, mx[0], mx[1], my},
                {mx[0], aKoef[0][0], aKoef[0][1], a[0]},
                {mx[1], aKoef[0][1], aKoef[1][1], a[1]},
                {mx[2], aKoef[0][2], aKoef[1][2], a[2]}
        };

        bArr[3] = determinant(matrixTemp5) / determinant(matrixTemp2);

        System.out.println("\nНатуралізоване рівняння регресії: ");
        System.out.printf("y = %.2f", bArr[0]);
        if (bArr[1] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[1]), symbols.get(dot), symbols.get(1));
        if (bArr[2] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[2]), symbols.get(dot), symbols.get(2));
        if (bArr[3] < 0) System.out.print(" - ");
        else System.out.print(" + ");
```

```
            System.out.printf("%.2f%sx%s\n", Math.abs(bArr[3]), symbols.get(dot), symbols.get(3));

            System.out.println("\nПеревірка: ");
            boolean ok = false;
            for (int i = 0; i < 4; i++) {
               ok = (float) (bArr[0] + bArr[1] * xArr[i][0] + bArr[2] * xArr[i][1] + bArr[3] * xArr[i][2]) == (float)
yAverage[i];
               System.out.printf("%.2f = %.2f\n", (bArr[0] + bArr[1] * xArr[i][0] + bArr[2] * xArr[i][1] + bArr[3] *
xArr[i][2]), yAverage[i]);
            }
            if (ok)
               System.out.printf("\nНатуралізовані коефіцієнти рівняння регресії b%s,b%s,b%s,b%s визначено
правильно\n",
                     symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3));
            else
               System.out.printf("\nНатуралізовані коефіцієнти рівняння регресії b%s,b%s,b%s,b%s визначено
неправильно\n",
                     symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3));

            double[] aNorm = new double[4];
            sum = 0;
            for (int i = 0; i < 4; i++) {
               sum += yAverage[i];
            }

            aNorm[0] = sum / 4.;
            aNorm[1] = bArr[1] * (MaxX1 - MinX1) / 2.;
            aNorm[2] = bArr[2] * (MaxX2 - MinX2) / 2.;
            aNorm[3] = bArr[3] * (MaxX3 - MinX3) / 2.;


            System.out.println("\nНормоване рівняння регресії: ");
            System.out.printf("y = %.2f", aNorm[0]);
            if (aNorm[1] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s", Math.abs(aNorm[1]), symbols.get(dot), symbols.get(1));
            if (aNorm[2] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s", Math.abs(aNorm[2]), symbols.get(dot), symbols.get(2));
            if (aNorm[3] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s\n", Math.abs(aNorm[3]), symbols.get(dot), symbols.get(3));


            System.out.println("\nПеревірка: ");
            for (int i = 0; i < 4; i++) {
               ok = (float) (aNorm[0] + aNorm[1] * x[i][1] + aNorm[2] * x[i][2] + aNorm[3] * x[i][3]) == (float)
yAverage[i];
               System.out.printf("%.2f = %.2f\n", (aNorm[0] + aNorm[1] * x[i][1] + aNorm[2] * x[i][2] + aNorm[3] *
x[i][3]), yAverage[i]);

            }
            if (ok) System.out.printf("\nНормовані коефіцієнти рівняння регресії a%s,a%s,a%s,a%s визначено
правильно\n",
                  symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3));
            else System.out.printf("\nНормовані коефіцієнти рівняння регресії a%s,a%s,a%s,a%s визначено
неправильно\n",
                  symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3));

            //критерій Кохрена

            for (int i = 0; i < 3; i++) {
               sum = 0;
```

```
                double[] yTemp = y.get(i);
                for (int j = 0; j < m; j++) {
                    sum += Math.pow((yTemp[j] - yAverage[i]), 2);
                }
                dispersionArr[i] = sum / m;
            }

            double maxDispersion = dispersionArr[0];
            for (int i = 0; i < 4; i++) {
                if (maxDispersion < dispersionArr[i]) maxDispersion = dispersionArr[i];
            }

            double Gp;
            sum = 0;
            for (int i = 0; i < 4; i++) {
                sum += dispersionArr[i];
            }
            Gp = maxDispersion / sum;

            f1 = m - 1;
            f2 = 4;
            q = 0.05;

            double[] KohrenTable = {0.9065, 0.7679, 0.6841, 0.6287, 0.5892, 0.5598, 0.5365, 0.5175, 0.5017, 0.4884,
0.4366, 0.372, 0.3093, 0.25};
            double Gt;

            if (f1 <= 1) Gt = KohrenTable[0];
            else if (f1 <= 2) Gt = KohrenTable[1];
            else if (f1 <= 3) Gt = KohrenTable[2];
            else if (f1 <= 4) Gt = KohrenTable[3];
            else if (f1 <= 5) Gt = KohrenTable[4];
            else if (f1 <= 6) Gt = KohrenTable[5];
            else if (f1 <= 7) Gt = KohrenTable[6];
            else if (f1 <= 8) Gt = KohrenTable[7];
            else if (f1 <= 9) Gt = KohrenTable[8];
            else if (f1 <= 10) Gt = KohrenTable[9];
            else if (f1 <= 16) Gt = KohrenTable[10];
            else if (f1 <= 36) Gt = KohrenTable[11];
            else if (f1 <= 144) Gt = KohrenTable[12];
            else Gt = KohrenTable[13];


            if (Gp < Gt) {
                System.out.printf("Gp = %.2f < Gt = %.2f\n", Gp, Gt);
                System.out.println("Дисперсії однорідні\n");
                work = false;
            } else {
                work = true;
                System.out.printf("Gp = %.2f > Gt = %.2f\n", Gp, Gt);
            }
            symbols.remove(symbols.size()-1);
            symbols.remove(symbols.size()-1);
            m++;
            symbols.add("\u2219");
            symbols.add("\u2260");
            if (work)
                System.out.println("ДИСПЕРСІЇ НЕОДНОРІДНІ\nПОМИЛКА : Gp > Gt \nЗБІЛЬШУЄМО
КІЛЬКІСТЬ ДОСЛІДІВ : m+1\n");
        }
        //критерій Стьюдента
        double Beta;
        double BetaKvadrat;
```

```
        double BetaKvadratAverage;

        int dot = symbols.size()-2;
        int not_eq = symbols.size()-1;

        sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += dispersionArr[i];
        }
        BetaKvadratAverage = sum / 4;
        BetaKvadrat = BetaKvadratAverage / (4. * m);
        Beta = Math.sqrt(BetaKvadrat);

        double[] beta = new double[4];
        for (int i = 0; i < 4; i++) {
            sum = 0;
            for (int j = 0; j < 4; j++) {
                sum += yAverage[j] * x[j][i];
            }
            beta[i] = sum / 4;
        }

        double[] t = new double[4];

        for (int i = 0; i < 4; i++) {
            t[i] = Math.abs(beta[i]) / Beta;
        }

        int f3 = f1 * f2;
        double[] studentTable = {2.306, 2.262, 2.228, 2.201, 2.179, 2.16, 2.145, 2.131, 2.12, 2.11, 2.101, 2.093,
2.086};
        if (f3 > 16) {
            System.out.println("Відсутнє значення для такого f3");
            System.exit(1);
        }
        double stNow = studentTable[f3 - 8];

        int d = 4;

        if (t[0] < stNow) {
            bArr[0] = 0;
            d--;
        }
        if (t[1] < stNow) {
            bArr[1] = 0;
            d--;
        }
        if (t[2] < stNow) {
            bArr[2] = 0;
            d--;
        }
        if (t[3] < stNow) {
            bArr[3] = 0;
            d--;
        }

        System.out.println("Рівняння регресії після критерія Стьюдента: ");
        System.out.printf("y = %.2f", bArr[0]);
        if (bArr[1] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[1]), symbols.get(dot), symbols.get(1));
        if (bArr[2] < 0) System.out.print(" - ");
        else System.out.print(" + ");
```

```
            System.out.printf("%.2f%sx%s", Math.abs(bArr[2]), symbols.get(dot), symbols.get(2));
            if (bArr[3] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s\n", Math.abs(bArr[3]), symbols.get(dot), symbols.get(3));

            double[] yAverageAfterStudent = new double[4];


            System.out.println("\nПеревірка: ");
            for (int i = 0; i < 4; i++) {
                System.out.printf("%.2f %s %.2f\n",
                        yAverageAfterStudent[i] = (bArr[0] + bArr[1] * xArr[i][0] + bArr[2] * xArr[i][1] + bArr[3] *
xArr[i][2]),
                        symbols.get(not_eq), yAverage[i]);
            }

            //критерій Фішера
            int f4 = 4 - d;
            double sKvadratAdekv;
            sum = 0;
            for (int i = 0; i < 4; i++) {
                sum += Math.pow(yAverageAfterStudent[i] - yAverage[i], 2);
            }
            sKvadratAdekv = sum * (double) (m / (4 - d));

            double Fp = sKvadratAdekv / BetaKvadratAverage;

            double[][] fisherTable = {
                    {5.3, 4.5, 4.1, 3.8, 3.7, 3.6, 3.3, 3.1, 2.9},
                    {4.8, 3.9, 3.5, 3.3, 3.1, 3.0, 2.7, 2.5, 2.3},
                    {4.5, 3.6, 3.2, 3.0, 2.9, 2.7, 2.4, 2.2, 2.0},
                    {4.4, 3.5, 3.1, 2.9, 2.7, 2.6, 2.3, 2.1, 1.9}
            };

            double fisherNow = 0;
            if (f4 <= 1) fisherNow = fisherTable[m - 3][0];
            else if (f4 <= 2) fisherNow = fisherTable[m - 3][1];
            else if (f4 <= 3) fisherNow = fisherTable[m - 3][2];
            else if (f4 <= 4) fisherNow = fisherTable[m - 3][3];
            if (Fp < fisherNow) {
                System.out.printf("\nFp = %.2f < Ft = %.2f\n", Fp, fisherNow);
            } else if (Fp > fisherNow) {
                System.out.printf("\nFp = %.2f > Ft = %.2f\n", Fp, fisherNow);
            }

            if (Fp > fisherNow) {
                System.out.println("\nРівняння регресії неадекватно оригіналу при q = 0.05");
                System.out.printf("Рівняння регресії з ефектом взаємодії має вигляд : y = b%s + b%s%sx%s +
b%s%sx%s + b%s%sx%s + " +
                        "b%s%s%sx%s%sx%s + b%s%s%sx%s%sx%s + b%s%s%sx%s%sx%s +
b%s%s%s%sx%s%sx%s%sx%s\n", symbols.get(0),
                        symbols.get(1), symbols.get(dot), symbols.get(1),
                        symbols.get(2), symbols.get(dot), symbols.get(2),
                        symbols.get(3), symbols.get(dot), symbols.get(3),
                        symbols.get(1), symbols.get(2), symbols.get(dot), symbols.get(1), symbols.get(dot), symbols.get(2),
                        symbols.get(1), symbols.get(3), symbols.get(dot), symbols.get(1), symbols.get(dot), symbols.get(3),
                        symbols.get(2), symbols.get(3), symbols.get(dot), symbols.get(2), symbols.get(dot), symbols.get(3),
                        symbols.get(1), symbols.get(2), symbols.get(3), symbols.get(dot), symbols.get(1), symbols.get(dot),
symbols.get(2), symbols.get(dot), symbols.get(3));

                double[][] xInteraction = {
                        {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},
            {1, 1, 1, 1, 1, 1, 1, 1}
    };

    double[][] xNaturInteraction = {
            {1, 10, 25, 40, 250, 400, 1000, 10000},
            {1, 10, 25, 45, 250, 450, 1125, 11250},
            {1, 10, 45, 40, 450, 400, 1800, 18000},
            {1, 10, 45, 45, 450, 450, 2025, 20250},
            {1, 40, 25, 40, 1000, 1600, 1000, 40000},
            {1, 40, 25, 45, 1000, 1800, 1125, 45000},
            {1, 40, 45, 40, 1800, 1600, 1800, 72000},
            {1, 40, 45, 45, 1800, 1800, 2025, 81000}
    };

    double[][] matrixTemp = new double[8][8];

    double[] kArr = new double[8];
    double[] yInteractionAverage = new double[8];
    double[] dispersionInteractionArr = new double[8];

    double[][] mCoefMatrixInteraction = new double[8][8];

    double[] bNatur = new double[8];
    double[] bNorm = new double[8];
    boolean workInteraction;
    m = 3;
    //while (workInteraction) {
    System.out.println("Нормована матриця планування експерименту з ефектом взаємодії: ");
    System.out.print("X0\tX1\tX2\tX3\tX1X2\tX1X3\tX2X3\tX1X2X3\t");
    for (int i = 1; i <= m; i++) {
       System.out.print("Y" + symbols.get(i) + "\t\t\t");
    }
    System.out.print("YAvr\t\t\tDisp");
    System.out.println();
    for (int i = 0; i < 8; i++) {
       double[] yTemp = new double[m];
       for (int j = 0; j < 8; j++) {
          if (xInteraction[i][j]>0){
             System.out.print(" " + (int) xInteraction[i][j]);
          }
          else System.out.print((int) xInteraction[i][j]);
          if (j < 4) System.out.print("\t");
          else System.out.print("\t\t");
       }
       for (int j = 0; j < m; j++) {
          yTemp[j] = (Math.random() * (MaxY - MinY)) + MinY;
          StringBuilder Y = new StringBuilder("" + (float) yTemp[j]);
          /* Дабы колонки были ровными, и текст не съезжал, я добавил проверку на количество
символов в строке:
          в конец каждого значения будут добавляться нули, чтоб ширина колонки была 9 символов*/
          if (Y.length() < 9){
             for (int k = Y.length(); k < 9; k++) {
                Y.append("0");
             }
             System.out.print(Y + "\t\t");
          }
          else System.out.print(Y + "\t\t");
       }
```

```
        sum = 0;
        for (int j = 0; j < m; j++) {
            sum += yTemp[j];
        }
        yInteractionAverage[i] = sum / m;
        System.out.print((float) yInteractionAverage[i] + "\t\t");
        sum = 0;
        for (int k = 0; k < m; k++) {
            sum += Math.pow((yTemp[k] - yInteractionAverage[i]), 2);
        }
        dispersionInteractionArr[i] = sum / m;
        System.out.println((float) dispersionInteractionArr[i]);
    }
    for (int i = 0; i < 8; i++) {
        for (int j = 0; j < 8; j++) {
            sum = 0;
            for (int k = 0; k < 8; k++) {

                sum += xNaturInteraction[k][i] * xNaturInteraction[k][j];
            }

            mCoefMatrixInteraction[i][j] = sum;
        }
    }
    for (int i = 0; i < 8; i++) {
        sum = 0;
        for (int j = 0; j < 8; j++) {
            sum += yInteractionAverage[j] * xNaturInteraction[j][i];
        }
        kArr[i] = sum;
    }
    double det = determinant(mCoefMatrixInteraction);
    for (int i = 0; i < 8; i++) {
        for (int j = 0; j < 8; j++) {
            for (int k = 0; k < 8; k++) {
                matrixTemp[j][k] = mCoefMatrixInteraction[j][k];
            }
        }
        for (int j = 0; j < 8; j++) {
            matrixTemp[j][i] = kArr[j];
        }
        bNatur[i] = determinant(matrixTemp) / det;
    }

    System.out.println("\nНатуралізоване рівняння регресії з ефектом взаємодії: ");
    System.out.printf("y = %.2f", bNatur[0]);
    if (bNatur[1] < 0) System.out.print(" - ");
    else System.out.print(" + ");
    System.out.printf("%.2f%sx%s", Math.abs(bNatur[1]), symbols.get(dot), symbols.get(1));
    if (bNatur[2] < 0) System.out.print(" - ");
    else System.out.print(" + ");
    System.out.printf("%.2f%sx%s", Math.abs(bNatur[2]), symbols.get(dot), symbols.get(2));
    if (bNatur[3] < 0) System.out.print(" - ");
    else System.out.print(" + ");
    System.out.printf("%.2f%sx%s", Math.abs(bNatur[3]), symbols.get(dot), symbols.get(3));
    if (bNatur[4] < 0) System.out.print(" - ");
    else System.out.print(" + ");
    System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNatur[4]), symbols.get(dot), symbols.get(1),
                                        symbols.get(dot), symbols.get(2));
    if (bNatur[5] < 0) System.out.print(" - ");
    else System.out.print(" + ");
    System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNatur[5]), symbols.get(dot), symbols.get(1),
                                        symbols.get(dot), symbols.get(3));
```

```
            if (bNatur[6] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNatur[6]), symbols.get(dot), symbols.get(2),
                                                symbols.get(dot), symbols.get(3));
            if (bNatur[7] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s%sx%s%sx%s\n", Math.abs(bNatur[7]), symbols.get(dot), symbols.get(1),
                                                    symbols.get(dot), symbols.get(2),
                                                    symbols.get(dot), symbols.get(3));

            System.out.println("\nПеревірка: ");
            boolean ok = false;
            for (int i = 0; i < 8; i++) {
               ok = (float) (bNatur[0] + bNatur[1] * xNaturInteraction[i][1] + bNatur[2] * xNaturInteraction[i][2]
                     + bNatur[3] * xNaturInteraction[i][3] + bNatur[4] * xNaturInteraction[i][4] + bNatur[5] *
xNaturInteraction[i][5]
                     + bNatur[6] * xNaturInteraction[i][6] + bNatur[7] * xNaturInteraction[i][7]) == (float)
yInteractionAverage[i];
               System.out.printf("%.2f = %.2f\n", (bNatur[0] + bNatur[1] * xNaturInteraction[i][1] + bNatur[2] *
xNaturInteraction[i][2]
                     + bNatur[3] * xNaturInteraction[i][3] + bNatur[4] * xNaturInteraction[i][4] + bNatur[5] *
xNaturInteraction[i][5]
                     + bNatur[6] * xNaturInteraction[i][6] + bNatur[7] * xNaturInteraction[i][7]),
yInteractionAverage[i]);
            }
            if (ok)
               System.out.printf("\nНатуралізовані коефіцієнти рівняння регресії
b%s,b%s,b%s,b%s,b%s%s,b%s%s,b%s%s,b%s%s%s визначено правильно\n",
                     symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3), symbols.get(1), symbols.get(2),
                     symbols.get(1), symbols.get(3), symbols.get(2), symbols.get(3),
                     symbols.get(1), symbols.get(2), symbols.get(3));
            else
               System.out.printf("\nНатуралізовані коефіцієнти рівняння регресії
b%s,b%s,b%s,b%s,b%s%s,b%s%s,b%s%s,b%s%s%s визначено неправильно\n",
                     symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3), symbols.get(1), symbols.get(2),
                     symbols.get(1), symbols.get(3), symbols.get(2), symbols.get(3),
                     symbols.get(1), symbols.get(2), symbols.get(3));

            for (int i = 0; i < 8; i++) {
               sum = 0;
               for (int j = 0; j < 8; j++) {
                  sum += yInteractionAverage[j] * xInteraction[j][i];
               }
               kArr[i] = sum;
            }

            for (int i = 0; i < 8; i++) {
               bNorm[i] = kArr[i] / 8.;
            }
            System.out.println("\nНормоване рівняння регресії з ефектом взаємодії: ");
            System.out.printf("y = %.2f", bNorm[0]);
            if (bNorm[1] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s", Math.abs(bNorm[1]), symbols.get(dot), symbols.get(1));
            if (bNorm[2] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s", Math.abs(bNorm[2]), symbols.get(dot), symbols.get(2));
            if (bNorm[3] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s", Math.abs(bNorm[3]), symbols.get(dot), symbols.get(3));
            if (bNorm[4] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNorm[4]), symbols.get(dot), symbols.get(1),
```

```
                                        symbols.get(dot), symbols.get(2));
            if (bNorm[5] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNorm[5]), symbols.get(dot), symbols.get(1),
                                        symbols.get(dot), symbols.get(3));
            if (bNorm[6] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNorm[6]), symbols.get(dot), symbols.get(2),
                                        symbols.get(dot), symbols.get(3));
            if (bNorm[7] < 0) System.out.print(" - ");
            else System.out.print(" + ");
            System.out.printf("%.2f%sx%s%sx%s%sx%s\n", Math.abs(bNorm[7]), symbols.get(dot), symbols.get(1),
                                            symbols.get(dot), symbols.get(2),
                                            symbols.get(dot), symbols.get(3));

            System.out.println("\nПеревірка: ");
            ok = false;
            for (int i = 0; i < 8; i++) {
               ok = (float) (bNorm[0] + bNorm[1] * xInteraction[i][1] + bNorm[2] * xInteraction[i][2]
                     + bNorm[3] * xInteraction[i][3] + bNorm[4] * xInteraction[i][4] + bNorm[5] * xInteraction[i][5]
                     + bNorm[6] * xInteraction[i][6] + bNorm[7] * xInteraction[i][7]) == (float)
yInteractionAverage[i];
               System.out.printf("%.2f = %.2f\n", (bNorm[0] + bNorm[1] * xInteraction[i][1] + bNorm[2] *
xInteraction[i][2]
                     + bNorm[3] * xInteraction[i][3] + bNorm[4] * xInteraction[i][4] + bNorm[5] * xInteraction[i][5]
                     + bNorm[6] * xInteraction[i][6] + bNorm[7] * xInteraction[i][7]), yInteractionAverage[i]);
            }
            if (ok)
               System.out.printf("\nНормовані коефіцієнти рівняння регресії
b%s,b%s,b%s,b%s,b%s%s,b%s%s,b%s%s,b%s%s%s визначено правильно\n",
                     symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3), symbols.get(1), symbols.get(2),
                     symbols.get(1), symbols.get(3), symbols.get(2), symbols.get(3),
                     symbols.get(1), symbols.get(2), symbols.get(3));
            else
               System.out.printf("\nНормовані коефіцієнти рівняння регресії
b%s,b%s,b%s,b%s,b%s%s,b%s%s,b%s%s,b%s%s%s визначено неправильно\n",
                     symbols.get(0), symbols.get(1), symbols.get(2), symbols.get(3), symbols.get(1), symbols.get(2),
                     symbols.get(1), symbols.get(3), symbols.get(2), symbols.get(3),
                     symbols.get(1), symbols.get(2), symbols.get(3));

            //критерій Кохрена

            double maxDispersionInteraction = dispersionInteractionArr[0];
            for (int i = 0; i < 4; i++) {
               if (maxDispersionInteraction < dispersionInteractionArr[i])
                  maxDispersionInteraction = dispersionInteractionArr[i];
            }

            double Gp;
            sum = 0;
            for (int i = 0; i < 4; i++) {
               sum += dispersionInteractionArr[i];
            }
            Gp = maxDispersionInteraction / sum;

            f1 = m - 1;
            f2 = 8;
            q = 0.05;

            double[] KohrenTableInteraction = {0.6798, 0.5157, 0.4377, 0.391, 0.3595, 0.3362, 0.3185, 0.3043,
0.2926, 0.2829, 0.2462, 0.2022, 0.1616, 0.125};
            double Gt;
```

```
            if (f1 <= 1) Gt = KohrenTableInteraction[0];
            else if (f1 <= 2) Gt = KohrenTableInteraction[1];
            else if (f1 <= 3) Gt = KohrenTableInteraction[2];
            else if (f1 <= 4) Gt = KohrenTableInteraction[3];
            else if (f1 <= 5) Gt = KohrenTableInteraction[4];
            else if (f1 <= 6) Gt = KohrenTableInteraction[5];
            else if (f1 <= 7) Gt = KohrenTableInteraction[6];
            else if (f1 <= 8) Gt = KohrenTableInteraction[7];
            else if (f1 <= 9) Gt = KohrenTableInteraction[8];
            else if (f1 <= 10) Gt = KohrenTableInteraction[9];
            else if (f1 <= 16) Gt = KohrenTableInteraction[10];
            else if (f1 <= 36) Gt = KohrenTableInteraction[11];
            else if (f1 <= 144) Gt = KohrenTableInteraction[12];
            else Gt = KohrenTableInteraction[13];


            if (Gp < Gt) {
                System.out.printf("Gp = %.2f < Gt = %.2f\n", Gp, Gt);
                System.out.println("Дисперсії однорідні\n");
                workInteraction = false;
            } else {
                workInteraction = true;
                System.out.printf("Gp = %.2f > Gt = %.2f\n", Gp, Gt);
            }
            m++;
            if (workInteraction) {
                System.out.println("ДИСПЕРСІЇ НЕОДНОРІДНІ\nПОМИЛКА : Gp > Gt \nЗБІЛЬШУЄМО
КІЛЬКІСТЬ ДОСЛІДІВ : m+1\n");
            }
            //}

            //критерій Стьюдента

            double sBetaKvadratAverageInteraction;
            double sBetaSInteraction;
            double sKvadratBetaSInteraction;
            sum = 0;
            for (int i = 0; i < 8; i++) {
                sum += dispersionInteractionArr[i];
            }
            sBetaKvadratAverageInteraction = sum / 8;
            sKvadratBetaSInteraction = sBetaKvadratAverageInteraction / (8. * m);
            sBetaSInteraction = Math.sqrt(sKvadratBetaSInteraction);

            double[] betaInteraction = new double[8];
            for (int i = 0; i < 8; i++) {
                sum = 0;
                for (int j = 0; j < 8; j++) {
                    sum += yInteractionAverage[j] * xInteraction[j][i];
                }
                betaInteraction[i] = sum / 8;
            }

            double[] tInteraction = new double[8];

            for (int i = 0; i < 8; i++) {
                tInteraction[i] = Math.abs(betaInteraction[i]) / sBetaSInteraction;
            }

            f3 = f1 * f2;
            double[] studentTableInteraction = {2.12, 2.11, 2.101, 2.093, 2.086, 2.08, 2.074, 2.069, 2.064, 2.06,
2.056};
            if (f3 > 24) {
```

```
    System.out.println("Відсутнє значення для такого f3");
    System.exit(1);
}
double stInteractionNow = studentTableInteraction[f3 - 16];


d = 8;

if (tInteraction[0] < stInteractionNow) {
    bNatur[0] = 0;
    d--;
}
if (tInteraction[1] < stInteractionNow) {
    bNatur[1] = 0;
    d--;
}
if (tInteraction[2] < stInteractionNow) {
    bNatur[2] = 0;
    d--;
}
if (tInteraction[3] < stInteractionNow) {
    bNatur[3] = 0;
    d--;
}
if (tInteraction[4] < stInteractionNow) {
    bNatur[4] = 0;
    d--;
}
if (tInteraction[5] < stInteractionNow) {
    bNatur[5] = 0;
    d--;
}
if (tInteraction[6] < stInteractionNow) {
    bNatur[6] = 0;
    d--;
}
if (tInteraction[7] < stInteractionNow) {
    bNatur[7] = 0;
    d--;
}

System.out.println("Рівняння регресії після критерію Стьюдента з ефектом взаємодії: ");
System.out.printf("y = %.2f", bNatur[0]);
if (bNatur[1] < 0) System.out.print(" - ");
else System.out.print(" + ");
System.out.printf("%.2f%sx%s", Math.abs(bNatur[1]), symbols.get(dot), symbols.get(1));
if (bNatur[2] < 0) System.out.print(" - ");
else System.out.print(" + ");
System.out.printf("%.2f%sx%s", Math.abs(bNatur[2]), symbols.get(dot), symbols.get(2));
if (bNatur[3] < 0) System.out.print(" - ");
else System.out.print(" + ");
System.out.printf("%.2f%sx%s", Math.abs(bNatur[3]), symbols.get(dot), symbols.get(3));
if (bNatur[4] < 0) System.out.print(" - ");
else System.out.print(" + ");
System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNatur[4]), symbols.get(dot), symbols.get(1),
        symbols.get(dot), symbols.get(2));
if (bNatur[5] < 0) System.out.print(" - ");
else System.out.print(" + ");
System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNatur[5]), symbols.get(dot), symbols.get(1),
        symbols.get(dot), symbols.get(3));
if (bNatur[6] < 0) System.out.print(" - ");
else System.out.print(" + ");
System.out.printf("%.2f%sx%s%sx%s", Math.abs(bNatur[6]), symbols.get(dot), symbols.get(2),
```

```
                symbols.get(dot), symbols.get(3));
        if (bNatur[7] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s%sx%s%sx%s\n", Math.abs(bNatur[7]), symbols.get(dot), symbols.get(1),
                symbols.get(dot), symbols.get(2),
                symbols.get(dot), symbols.get(3));

        double[] yAverageAfterStudentInteraction = new double[8];


        System.out.println("\nПеревірка: ");
        for (int i = 0; i < 8; i++) {
            System.out.printf("%.2f %s %.2f\n", yAverageAfterStudentInteraction[i] = (bNatur[0] + bNatur[1] *
xNaturInteraction[i][1] + bNatur[2] * xNaturInteraction[i][2]
                    + bNatur[3] * xNaturInteraction[i][3] + bNatur[4] * xNaturInteraction[i][4] + bNatur[5] *
xNaturInteraction[i][5]
                    + bNatur[6] * xNaturInteraction[i][6] + bNatur[7] * xNaturInteraction[i][7]), symbols.get(not_eq),
yInteractionAverage[i]);
        }

        //критерій Фішера

        f4 = 8 - d;
        double sKvadratAdekvInteraction;
        sum = 0;
        for (int i = 0; i < 8; i++) {
            sum += Math.pow(yAverageAfterStudentInteraction[i] - yInteractionAverage[i], 2);
        }
        sKvadratAdekvInteraction = sum * (m / (double) (8 - d));

        double FpInteraction = sKvadratAdekvInteraction / sBetaKvadratAverageInteraction;

        double[][] fisherTableInteraction = {
                {4.5, 3.6, 3.2, 3.0, 2.9, 2.7, 2.4, 2.2, 2.0},
                {4.3, 3.4, 3.0, 2.8, 2.6, 2.5, 2.2, 2.0, 1.7},
                {4.1,3.2,2.9,2.6,2.5,2.3,2.0,1.8,1.5}
        };

        double fisherIntercationNow = 0;
        if (f4 <= 1) fisherIntercationNow = fisherTableInteraction[m - 3][0];
        else if (f4 <= 2) fisherIntercationNow = fisherTableInteraction[m - 3][1];
        else if (f4 <= 3) fisherIntercationNow = fisherTableInteraction[m - 3][2];
        else if (f4 <= 4) fisherIntercationNow = fisherTableInteraction[m - 3][3];
        else if (f4 <= 5) fisherIntercationNow = fisherTableInteraction[m - 3][4];
        else if (f4 <= 6) fisherIntercationNow = fisherTableInteraction[m - 3][5];
        else if (f4 <= 12) fisherIntercationNow = fisherTableInteraction[m - 3][6];
        if (FpInteraction < fisherIntercationNow) {
            System.out.printf("\nFp = %.2f < Ft = %.2f\n", FpInteraction, fisherIntercationNow);
        } else if (FpInteraction > fisherIntercationNow) {
            System.out.printf("\nFp = %.2f > Ft = %.2f\n", FpInteraction, fisherIntercationNow);
        }

        if (FpInteraction > fisherIntercationNow) {
            System.out.println("\nРівняння регресії з ефектом взаємодії неадекватно оригіналу при q = 0.05");
            m =3;
            work  = true;

        } else if (FpInteraction < fisherIntercationNow) {
            System.out.println("\nРівняння регресії з ефектом взаємодії адекватно оригіналу при q = 0.05");
            restart = false;
        }
    } else {
        System.out.println("\nРівняння регресії адекватно оригіналу при q = 0.05");
```

```
            restart = false;
        }
    }
  }
}
```

Результат роботи:

```
"C:\Program Files\JetBrains\IntelliJ IDEA 2020.1.1\jbr\bin\java.exe" "-javaagent:C:\Program Files\
Задайте значення m:
3
Лінійне рівняння регресії для нормованих значень x має вигляд : y = b0 + b1·x1 + b2·x2 + b3·x3

Нормована матриця планування експерименту :
X0  X1  X2  X3  Y1              Y2              Y3
 1  -1  -1  -1  233.18100       237.46297       229.32600
 1  -1   1   1  226.72655       229.43634       242.64203
 1   1  -1   1  235.40690       229.42258       240.85071
 1   1   1  -1  227.48097       238.39806       242.69629

Матриця планування експерименту :
X1  X2  X3  Y1              Y2              Y3
10  25  40  233.18100       237.46297       229.32600
10  45  45  226.72655       229.43634       242.64203
40  25  45  235.40690       229.42258       240.85071
40  45  40  227.48097       238.39806       242.69629

Натуралізоване рівняння регресії:
y = 237,52 + 0,09·x1 + 0,01·x2 - 0,14·x3

Перевірка:
233,32 = 233,32
232,93 = 232,93
235,23 = 235,23
236,19 = 236,19

Натуралізовані коефіцієнти рівняння регресії b0,b1,b2,b3 визначено правильно
```

```
Натуралізовані коефіцієнти рівняння регресії b0,b1,b2,b3 визначено правильно

Нормоване рівняння регресії:
y = 234,42 + 1,29·x1 + 0,14·x2 - 0,34·x3

Перевірка:
233,32 = 233,32
232,93 = 232,93
235,23 = 235,23
236,19 = 236,19

Нормовані коефіцієнти рівняння регресії a0,a1,a2,a3 визначено правильно
Gp = 0,60 < Gt = 0,77
Дисперсії однорідні

Рівняння регресії після критерія Стьюдента:
y = 237,52 + 0,00·x1 + 0,00·x2 + 0,00·x3

Перевірка:
237,52 ≠ 233,32
237,52 ≠ 232,93
237,52 ≠ 235,23
237,52 ≠ 236,19

Fp = 2,25 < Ft = 3,50

Рівняння регресії адекватно оригіналу при q = 0.05

Process finished with exit code 0
```

Висновки:

Під час виконання лабораторної роботи було змодельовано трьохфакторний експеримент при використанні лінійного рівняння регресії та рівняння регресії з ефектом взаємодії, складено матрицю планування експерименту, було визначено коефіцієнти рівняння регресії(натуралізовані та нормовані), виконано перевірку правильності розрахунку коефіцієнтів рівняння регресії. Також було проведено 3 статистичні перевірки(використання критеріїв Кохрена, Стьюдента та Фішера). При виявленні неадекватності лінійного рівняння регресії оригіналу було застосовано ефект взаємодії факторів. Довірча ймовірність в даній роботі дорівнює 0.95, відповідно рівень значимості q = 0.05.