According to the question, My problem was from suguru_book_3 and the problem no was 1.

```
import org.chocosolver.solver.Model;
import org.chocosolver.solver.Solver;
import org.chocosolver.solver.variables.IntVar;

public class Suguru {
    public static void main(String[] args) {

        int i, j, k;

        // 1. Create a Model
        Model model = new Model("Assignment of Suguru by 201914032");
        IntVar[][] bd = model.intVarMatrix("bd", 6, 6, 1, 5);

        // initializing the container/boxes to fit into the board's containers
        IntVar[] b0 = model.intVarArray("b0", 1, 1, 1);
        IntVar[] b1 = model.intVarArray("b1", 5, 1, 5);
        IntVar[] b2 = model.intVarArray("b2", 5, 1, 5);
        IntVar[] b3 = model.intVarArray("b3", 5, 1, 5);
        IntVar[] b4 = model.intVarArray("b4", 5, 1, 5);
        IntVar[] b5 = model.intVarArray("b5", 5, 1, 5);
        IntVar[] b6 = model.intVarArray("b6", 5, 1, 5);
        IntVar[] b7 = model.intVarArray("b7", 5, 1, 5);

        /* post constraints for the given hints or clues */

        model.arithm (bd[0][4], "=", 3).post();
        model.arithm (bd[1][2], "=", 2).post();
        model.arithm (bd[2][5], "=", 5).post();
        model.arithm (bd[3][2], "=", 4).post();
        model.arithm (bd[4][1], "=", 5).post();
        model.arithm (bd[4][5], "=", 1).post();
        model.arithm (bd[5][1], "=", 4).post();

        /* each box variable is associated with appropriate cell positions in board */

        model.arithm (bd[0][0], "=", b0[0]).post();
        model.arithm (bd[0][1], "=", b1[0]).post();
        model.arithm (bd[1][1], "=", b1[1]).post();
        model.arithm (bd[2][1], "=", b1[2]).post();
        model.arithm (bd[1][2], "=", b1[3]).post();
        model.arithm (bd[1][3], "=", b1[4]).post();

        model.arithm (bd[0][2], "=", b2[0]).post();
        model.arithm (bd[0][3], "=", b2[1]).post();
        model.arithm (bd[0][4], "=", b2[2]).post();
        model.arithm (bd[0][5], "=", b2[3]).post();
        model.arithm (bd[1][5], "=", b2[4]).post();

        model.arithm (bd[1][0], "=", b3[0]).post();
        model.arithm (bd[2][0], "=", b3[1]).post();
        model.arithm (bd[3][0], "=", b3[2]).post();
        model.arithm (bd[4][0], "=", b3[3]).post();
        model.arithm (bd[4][1], "=", b3[4]).post();
    model.arithm (bd[2][2], "=", b4[0]).post();
    model.arithm (bd[3][2], "=", b4[1]).post();
    model.arithm (bd[4][2], "=", b4[2]).post();
    model.arithm (bd[3][1], "=", b4[3]).post();
    model.arithm (bd[3][3], "=", b4[4]).post();

    model.arithm (bd[1][4], "=", b5[0]).post();
    model.arithm (bd[2][4], "=", b5[1]).post();
    model.arithm (bd[3][4], "=", b5[2]).post();
    model.arithm (bd[2][3], "=", b5[3]).post();
    model.arithm (bd[2][5], "=", b5[4]).post();

    model.arithm (bd[5][0], "=", b6[0]).post();
    model.arithm (bd[5][1], "=", b6[1]).post();
    model.arithm (bd[5][2], "=", b6[2]).post();
    model.arithm (bd[5][3], "=", b6[3]).post();
    model.arithm (bd[4][3], "=", b6[4]).post();

    model.arithm (bd[4][4], "=", b7[0]).post();
    model.arithm (bd[4][5], "=", b7[1]).post();
    model.arithm (bd[5][4], "=", b7[2]).post();
    model.arithm (bd[5][5], "=", b7[3]).post();
    model.arithm (bd[3][5], "=", b7[4]).post();

// this nested loop ensures that the adjacent cells contain different values
for ( i = 0; i < 6; i++){
    for ( j = 0; j < 6; j++){

        IntVar[] d0 = model.intVarArray("d0", 2, 1, 5);
        IntVar[] d1 = model.intVarArray("d1", 2, 1, 5);
        IntVar[] d2 = model.intVarArray("d2", 2, 1, 5);
        IntVar[] d3 = model.intVarArray("d3", 2, 1, 5);

        IntVar[] d4 = model.intVarArray("d4", 2, 1, 5);
        IntVar[] d5 = model.intVarArray("d5", 2, 1, 5);
        IntVar[] d6 = model.intVarArray("d6", 2, 1, 5);
        IntVar[] d7 = model.intVarArray("d7", 2, 1, 5);

        if(i-1>=0)
        {
            model.arithm (bd[i-1][j], "=", d0[0]).post();
            model.arithm (bd[i][j], "=", d0[1]).post();
            model.allDifferent(d0).post();
        }
        if(i+1<6)
        {
            model.arithm (bd[i+1][j], "=", d1[0]).post();
            model.arithm (bd[i][j], "=", d1[1]).post();
            model.allDifferent(d1).post();
        }
```

```
            if(j-1>=0)
            {
                model.arithm (bd[i][j-1], "=", d2[0]).post();
                model.arithm (bd[i][j], "=", d2[1]).post();
                model.allDifferent(d2).post();
            }
            if(j+1<6)
            {
                model.arithm (bd[i][j+1], "=", d3[0]).post();
                model.arithm (bd[i][j], "=", d3[1]).post();
                model.allDifferent(d3).post();
            }
            if(i-1>=0 && j-1>=0)
            {
                model.arithm (bd[i-1][j-1], "=", d4[0]).post();
                model.arithm (bd[i][j], "=", d4[1]).post();
                model.allDifferent(d4).post();
            }
            if(i+1<6 && j-1>=0)
            {
                model.arithm (bd[i+1][j-1], "=", d5[0]).post();
                model.arithm (bd[i][j], "=", d5[1]).post();
                model.allDifferent(d5).post();
            }
            if(i-1>=0 && j+1<6)
            {
                model.arithm (bd[i-1][j+1], "=", d6[0]).post();
                model.arithm (bd[i][j], "=", d6[1]).post();
                model.allDifferent(d6).post();
            }
            if(i+1<6 && j+1<6)
            {
                model.arithm (bd[i+1][j+1], "=", d7[0]).post();
                model.arithm (bd[i][j], "=", d7[1]).post();
                model.allDifferent(d7).post();
            }

        }
    }




        // ensuring all the numbers in each containers are different
        model.allDifferent(b0).post();
        model.allDifferent(b1).post();
        model.allDifferent(b2).post();
        model.allDifferent(b3).post();
        model.allDifferent(b4).post();
        model.allDifferent(b5).post();
        model.allDifferent(b6).post();
        model.allDifferent(b7).post();

        // Solve the problem
        Solver solver = model.getSolver();
        solver.showStatistics();
        solver.showSolutions();
        solver.findSolution();

        // 5. Print the solution
        for ( i = 0; i < 6; i++){
            for ( j = 0; j < 6; j++){

                System.out.print(" ");
                /* get the value for the board position [i][j] for the solved board */
                k = bd [i][j].getValue();
                System.out.print(k );
            }
            System.out.println();
        }

    }

}
```