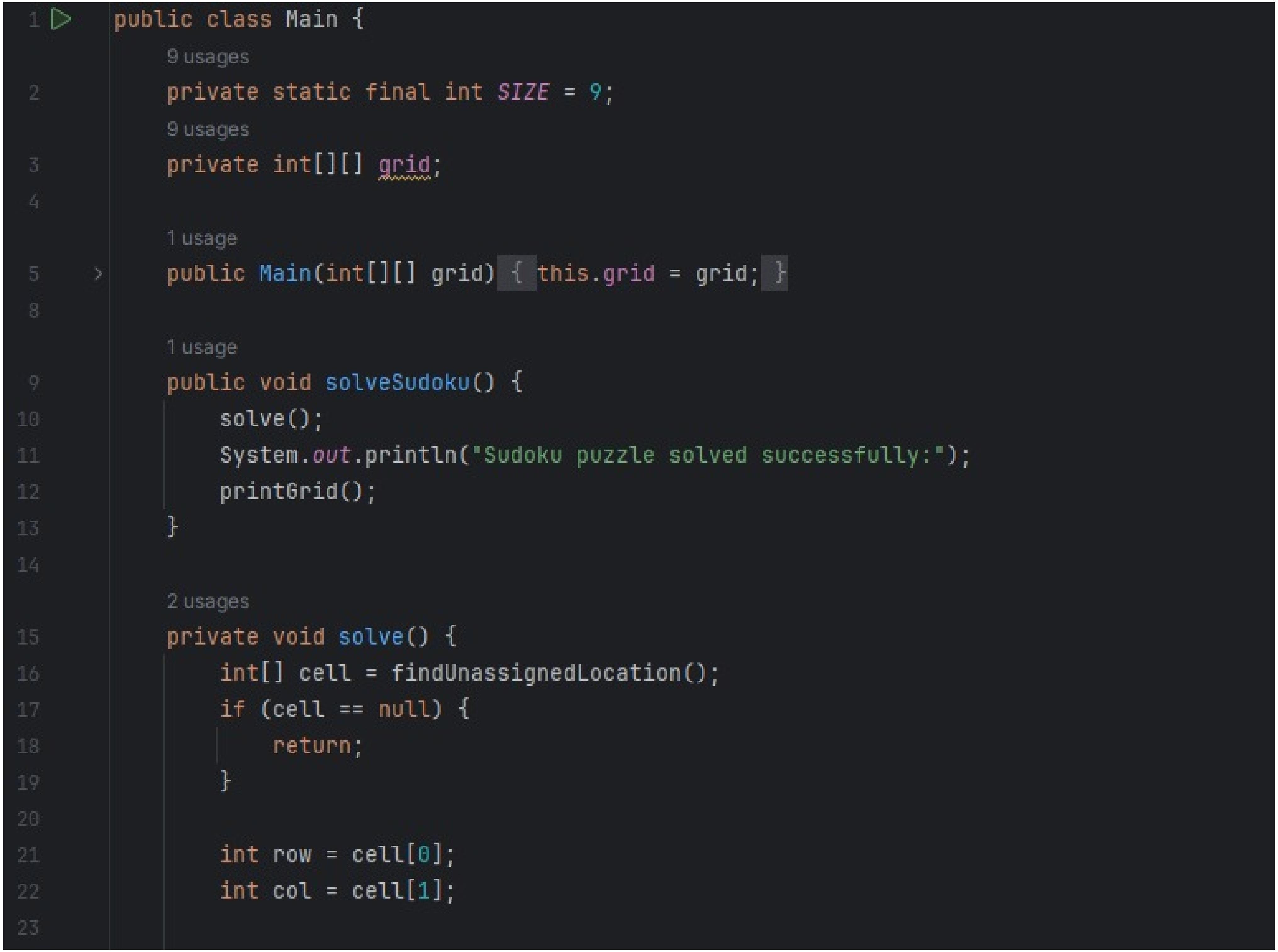
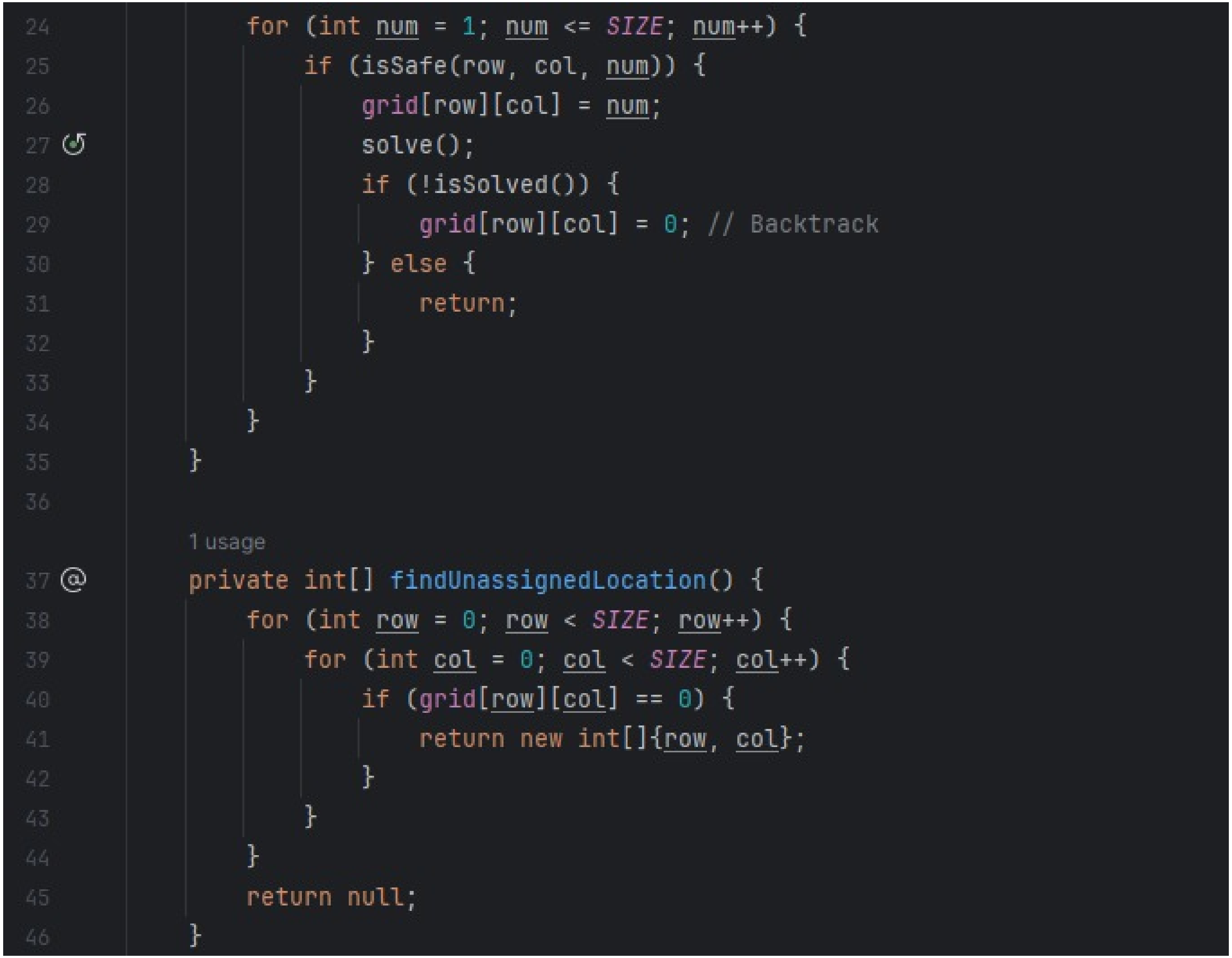
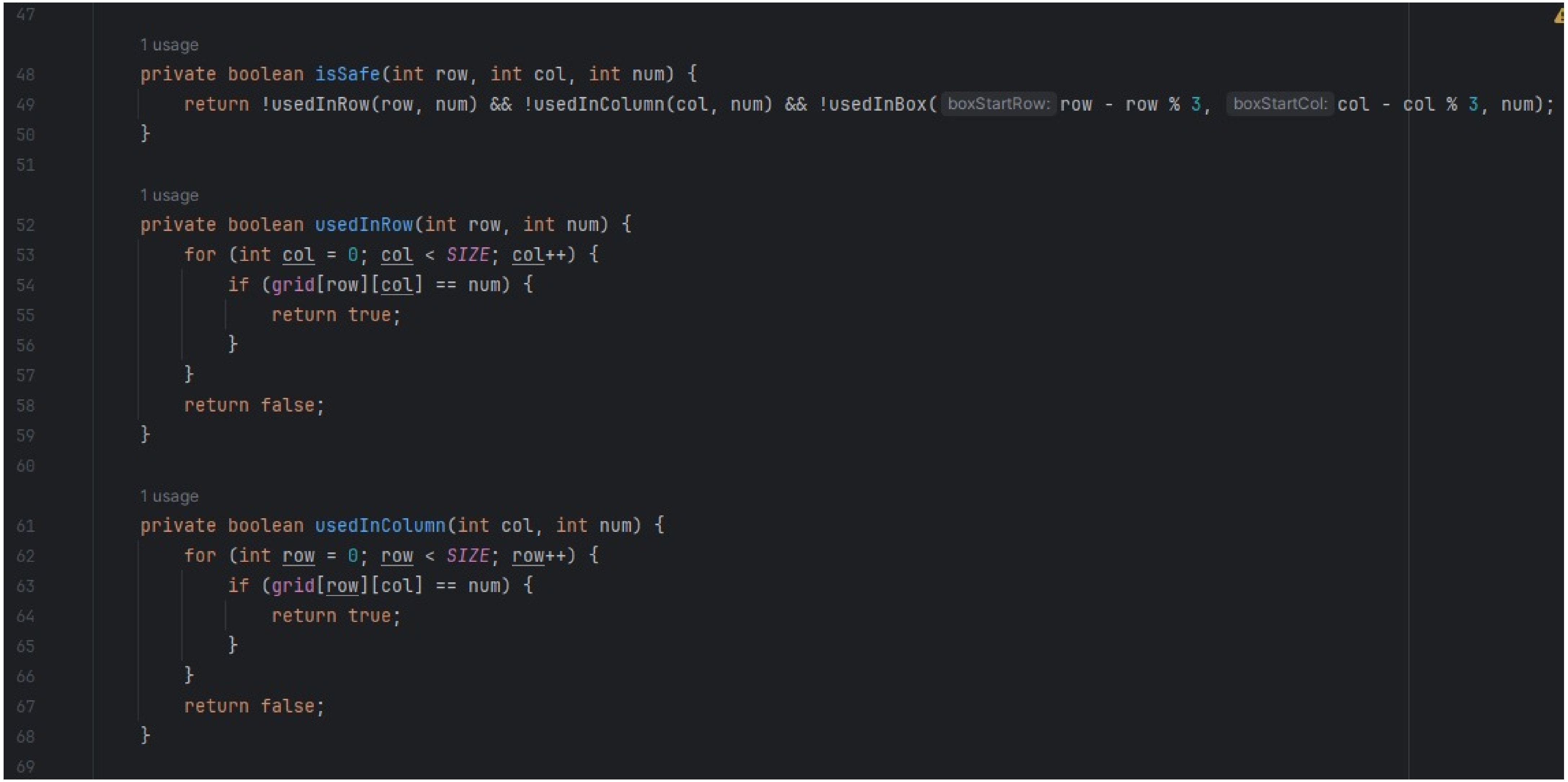
INPUT



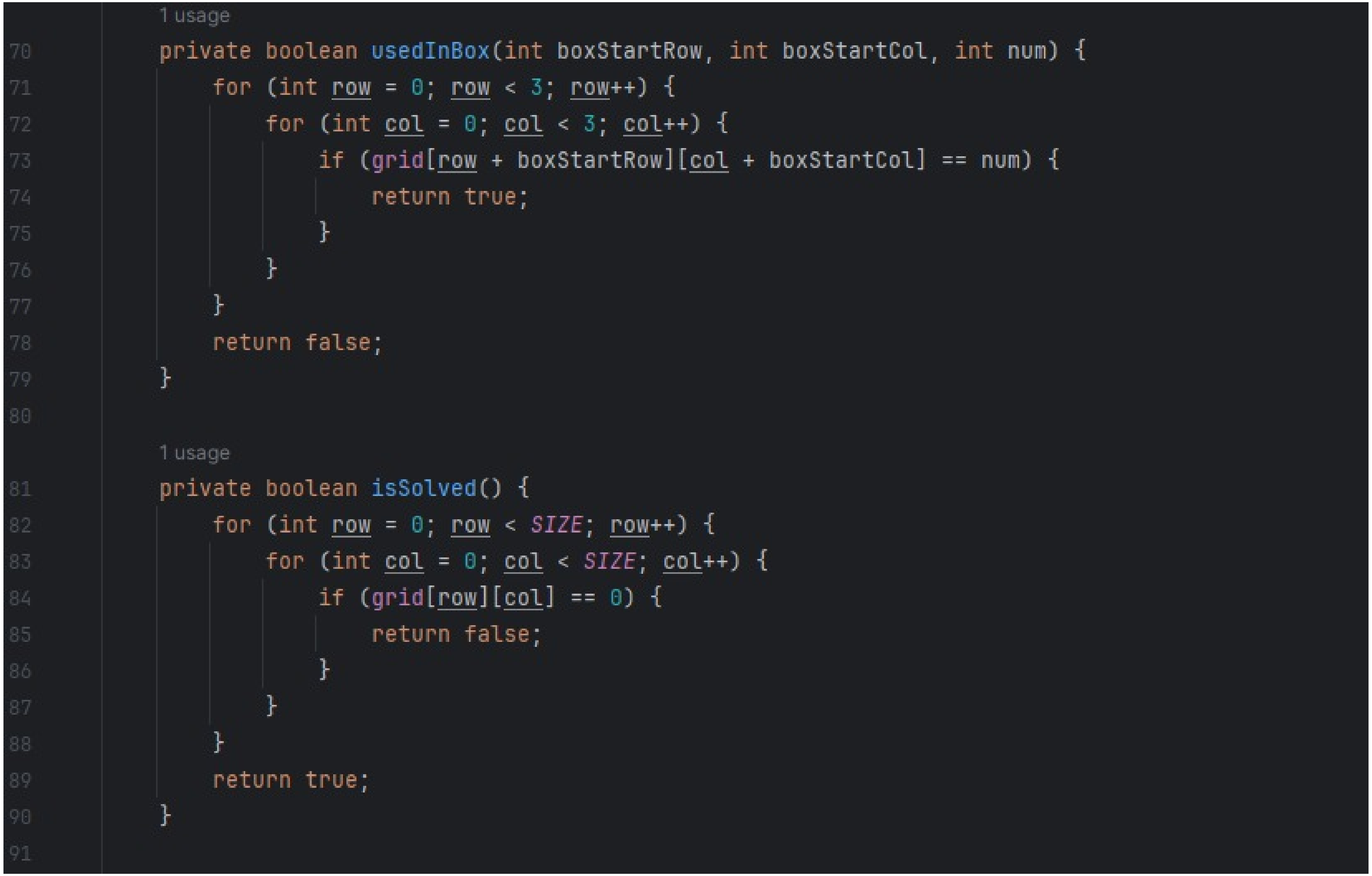
```java
public class Main {
    9 usages
    private static final int SIZE = 9;
    9 usages
    private int[][] grid;

    1 usage
    public Main(int[][] grid) { this.grid = grid; }

    1 usage
    public void solveSudoku() {
        solve();
        System.out.println("Sudoku puzzle solved successfully:");
        printGrid();
    }

    2 usages
    private void solve() {
        int[] cell = findUnassignedLocation();
        if (cell == null) {
            return;
        }

        int row = cell[0];
        int col = cell[1];

```



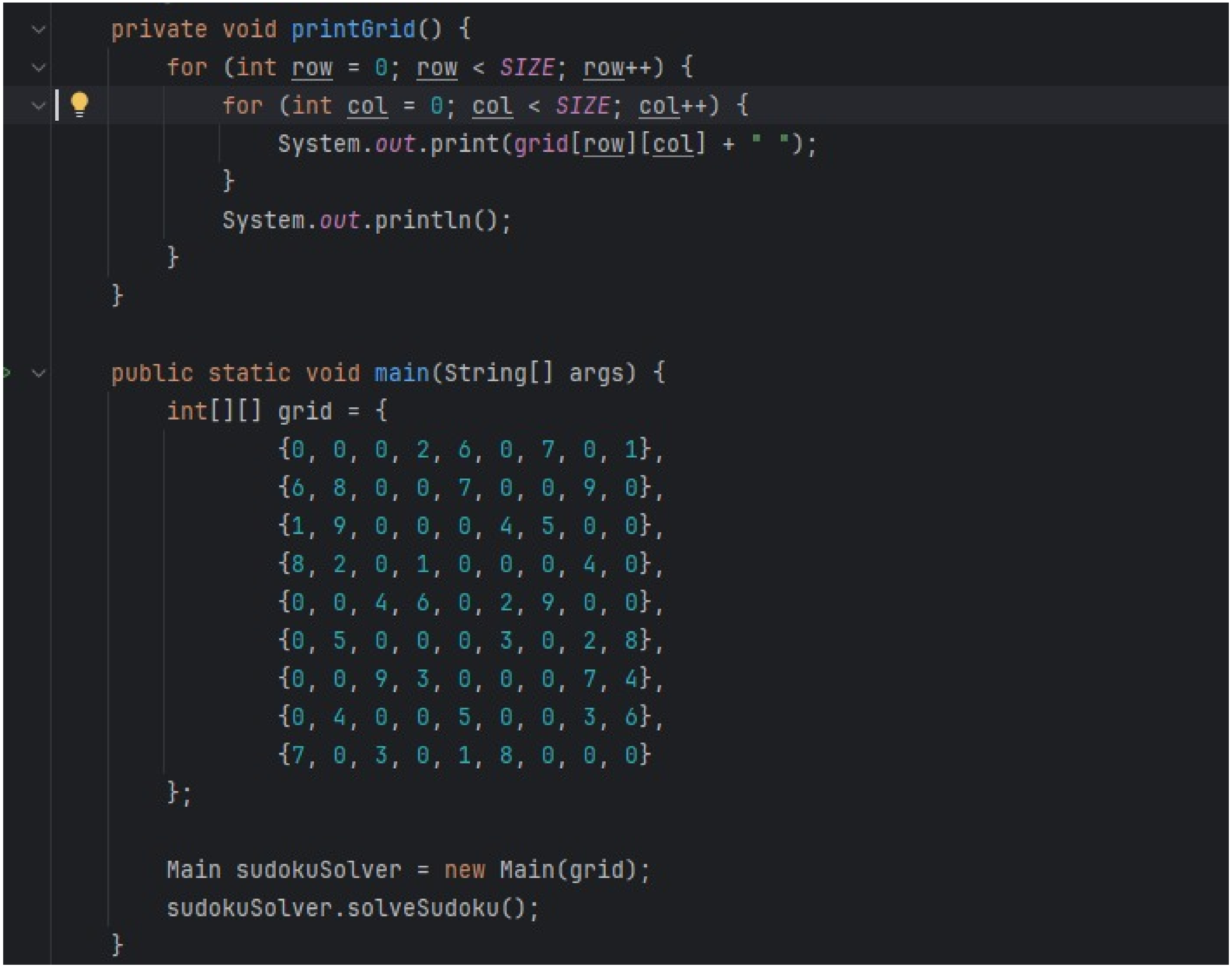
```java
        for (int num = 1; num <= SIZE; num++) {
            if (isSafe(row, col, num)) {
                grid[row][col] = num;
                solve();
                if (!isSolved()) {
                    grid[row][col] = 0; // Backtrack
                } else {
                    return;
                }
            }
        }
    }

    private int[] findUnassignedLocation() {
        for (int row = 0; row < SIZE; row++) {
            for (int col = 0; col < SIZE; col++) {
                if (grid[row][col] == 0) {
                    return new int[]{row, col};
                }
            }
        }
        return null;
    }
```



```java

    1 usage
    private boolean isSafe(int row, int col, int num) {
        return !usedInRow(row, num) && !usedInColumn(col, num) && !usedInBox( boxStartRow: row - row % 3,  boxStartCol: col - col % 3, num);
    }

    1 usage
    private boolean usedInRow(int row, int num) {
        for (int col = 0; col < SIZE; col++) {
            if (grid[row][col] == num) {
                return true;
            }
        }
        return false;
    }

    1 usage
    private boolean usedInColumn(int col, int num) {
        for (int row = 0; row < SIZE; row++) {
            if (grid[row][col] == num) {
                return true;
            }
        }
        return false;
    }

```

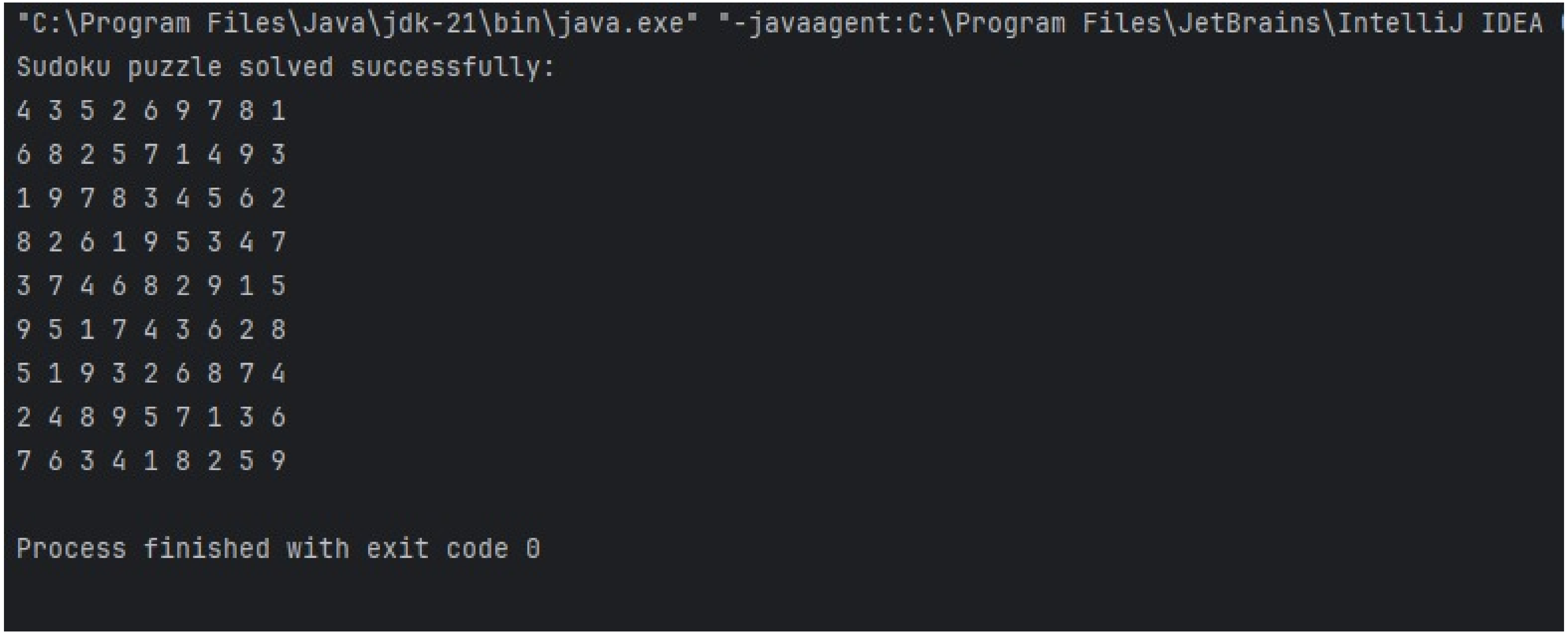


```java
    1 usage
    private boolean usedInBox(int boxStartRow, int boxStartCol, int num) {
        for (int row = 0; row < 3; row++) {
            for (int col = 0; col < 3; col++) {
                if (grid[row + boxStartRow][col + boxStartCol] == num) {
                    return true;
                }
            }
        }
        return false;
    }

    1 usage
    private boolean isSolved() {
        for (int row = 0; row < SIZE; row++) {
            for (int col = 0; col < SIZE; col++) {
                if (grid[row][col] == 0) {
                    return false;
                }
            }
        }
        return true;
    }

```



```java
    private void printGrid() {
        for (int row = 0; row < SIZE; row++) {
            for (int col = 0; col < SIZE; col++) {
                System.out.print(grid[row][col] + " ");
            }
            System.out.println();
        }
    }

    public static void main(String[] args) {
        int[][] grid = {
                {0, 0, 0, 2, 6, 0, 7, 0, 1},
                {6, 8, 0, 0, 7, 0, 0, 9, 0},
                {1, 9, 0, 0, 0, 4, 5, 0, 0},
                {8, 2, 0, 1, 0, 0, 0, 4, 0},
                {0, 0, 4, 6, 0, 2, 9, 0, 0},
                {0, 5, 0, 0, 0, 3, 0, 2, 8},
                {0, 0, 9, 3, 0, 0, 0, 7, 4},
                {0, 4, 0, 0, 5, 0, 0, 3, 6},
                {7, 0, 3, 0, 1, 8, 0, 0, 0}
        };

        Main sudokuSolver = new Main(grid);
        sudokuSolver.solveSudoku();
    }
```

OUTPUT



```
"C:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA
Sudoku puzzle solved successfully:
4 3 5 2 6 9 7 8 1
6 8 2 5 7 1 4 9 3
1 9 7 8 3 4 5 6 2
8 2 6 1 9 5 3 4 7
3 7 4 6 8 2 9 1 5
9 5 1 7 4 3 6 2 8
5 1 9 3 2 6 8 7 4
2 4 8 9 5 7 1 3 6
7 6 3 4 1 8 2 5 9

Process finished with exit code 0
```