DÉMINEUR EN GO

Ronan HENRY
Sully GRESSUS
Johan LAHOUGUE

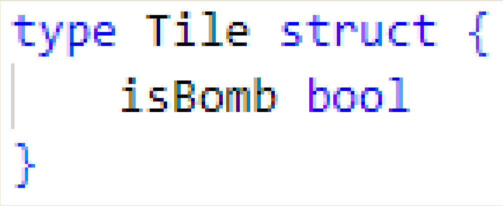
PROBLÉMATIQUE

COMMENT GÉNÉRER ET RÉSOUDRE EFFICACEMENT UNE GRILLE DE DÉMINEUR ?

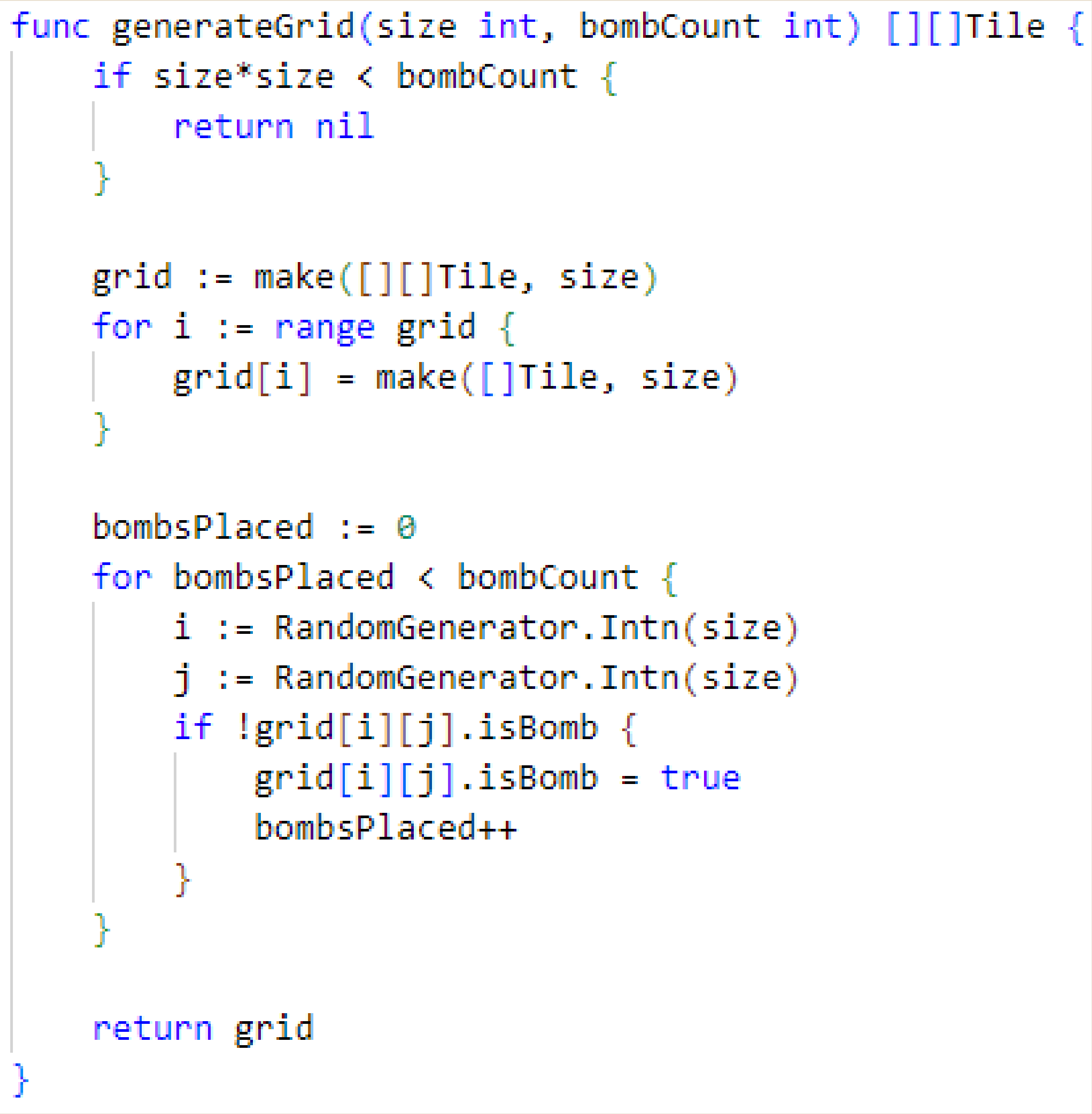
L'APPROCHE NAÏVE

- GÉNÉRER UNE GRILLE DE DÉMINEUR (25X25, 75 BOMBES)
- RÉSOUDRE LA GRILLE

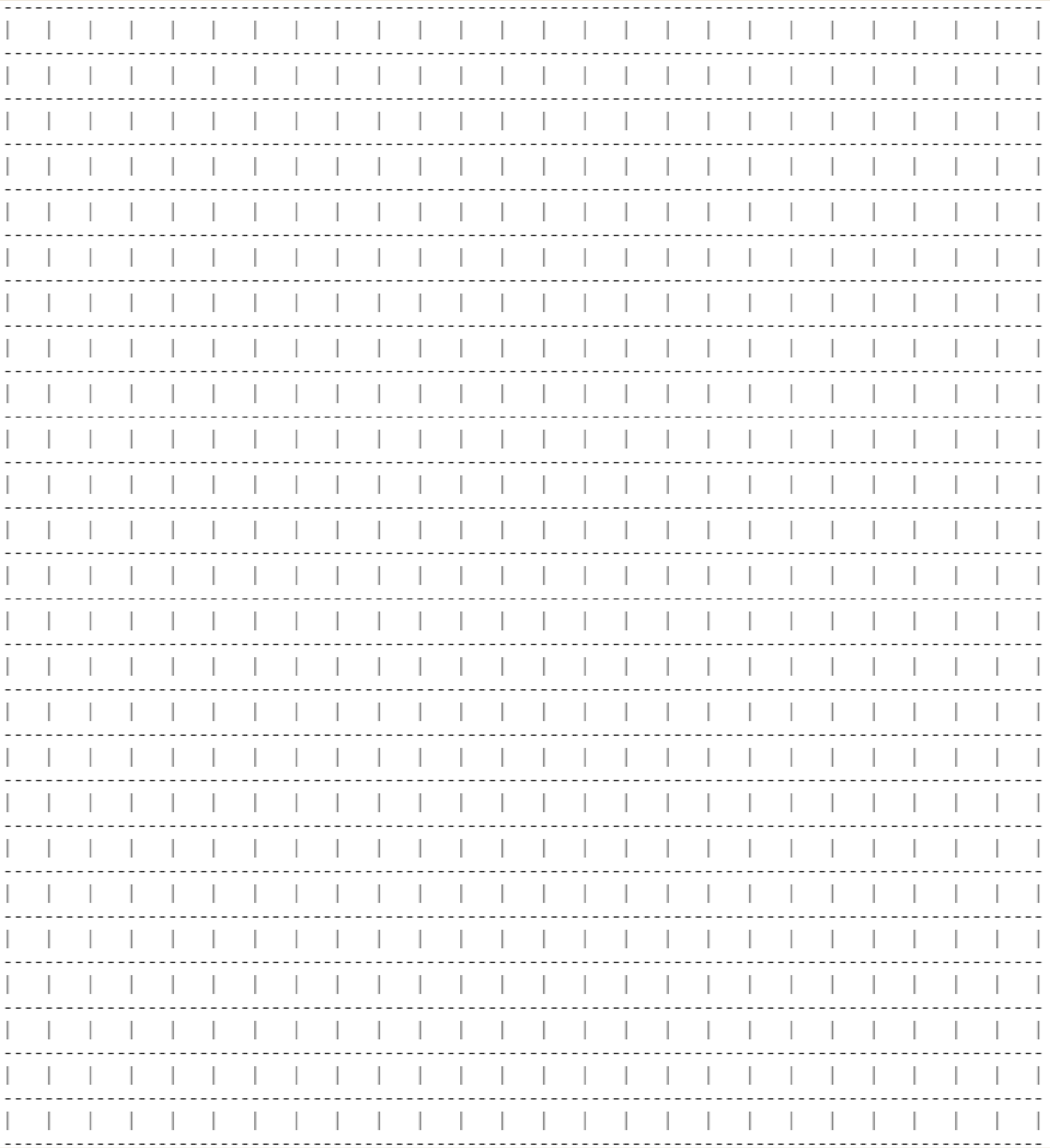
GÉNÉRATION DE LA GRILLE



```go
type Tile struct {
    isBomb bool
}
```

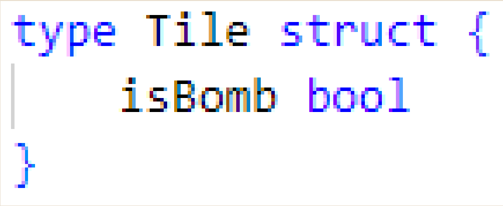


```go
func generateGrid(size int, bombCount int) [][]Tile {
    if size*size < bombCount {
        return nil
    }

    grid := make([][]Tile, size)
    for i := range grid {
        grid[i] = make([]Tile, size)
    }

    bombsPlaced := 0
    for bombsPlaced < bombCount {
        i := RandomGenerator.Intn(size)
        j := RandomGenerator.Intn(size)
        if !grid[i][j].isBomb {
            grid[i][j].isBomb = true
            bombsPlaced++
        }
    }

    return grid
}
```

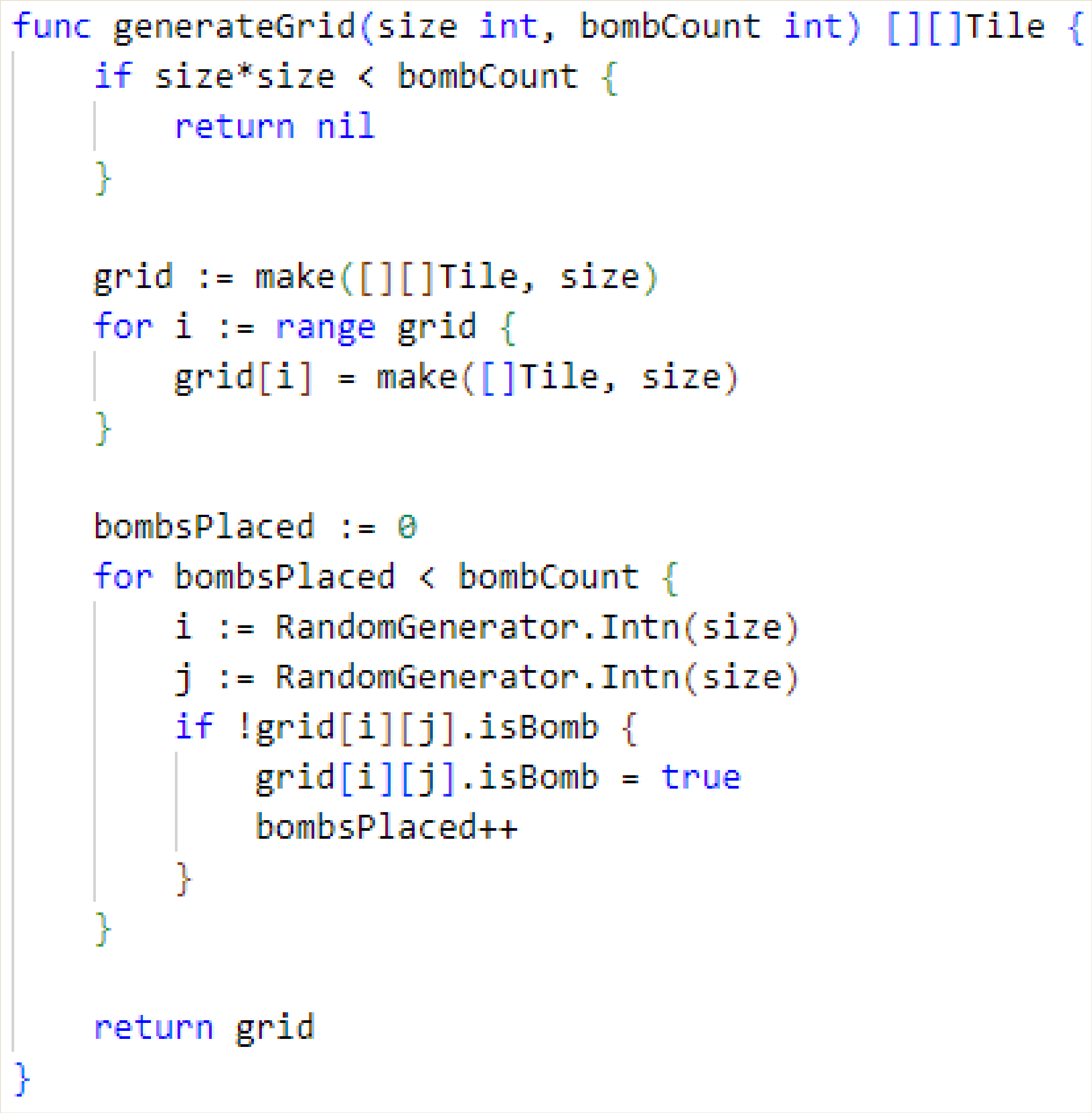


```
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
```

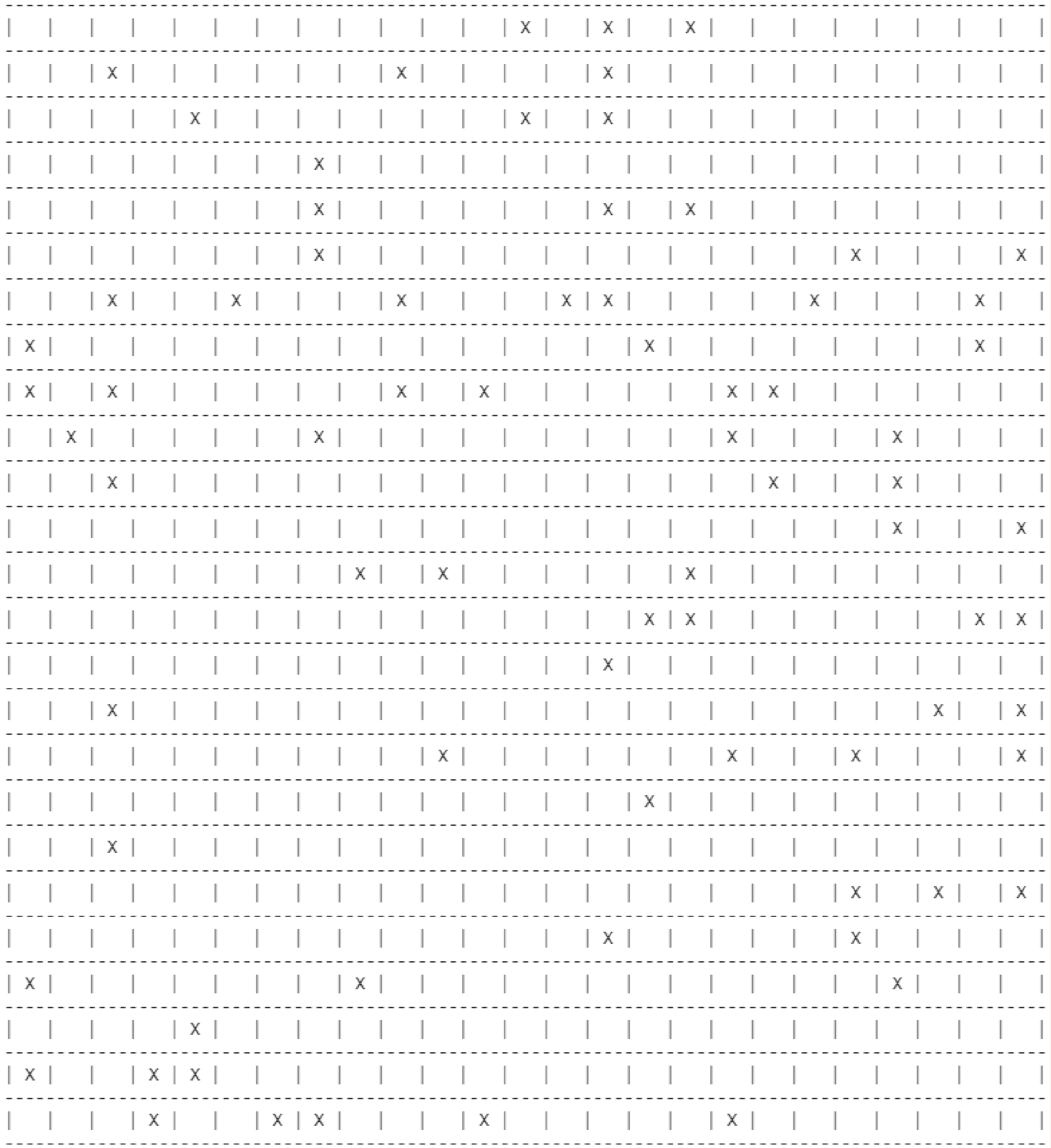
GÉNÉRATION DE LA GRILLE



```go
type Tile struct {
    isBomb bool
}
```

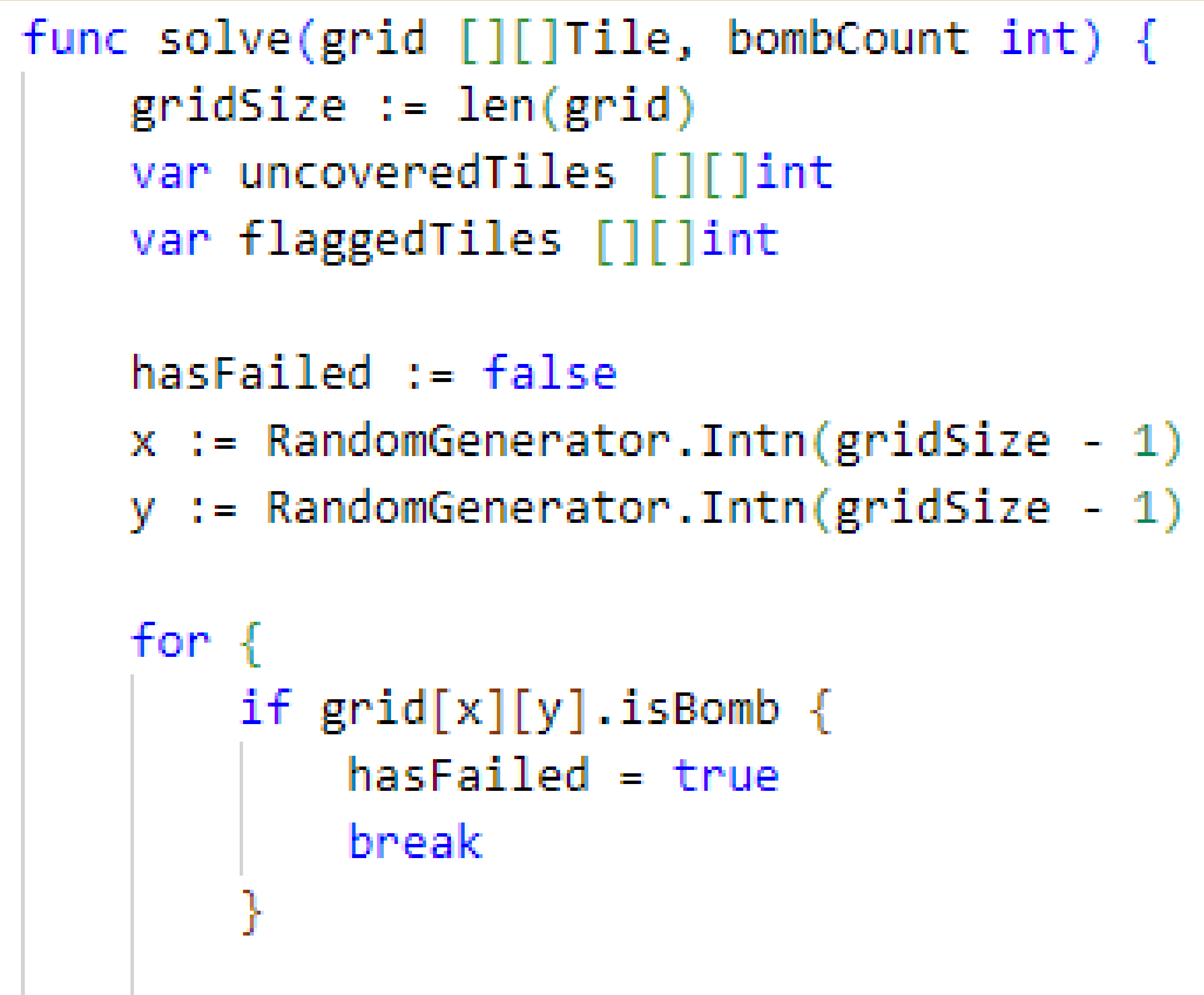


```go
func generateGrid(size int, bombCount int) [][]Tile {
    if size*size < bombCount {
        return nil
    }

    grid := make([][]Tile, size)
    for i := range grid {
        grid[i] = make([]Tile, size)
    }

    bombsPlaced := 0
    for bombsPlaced < bombCount {
        i := RandomGenerator.Intn(size)
        j := RandomGenerator.Intn(size)
        if !grid[i][j].isBomb {
            grid[i][j].isBomb = true
            bombsPlaced++
        }
    }

    return grid
}
```

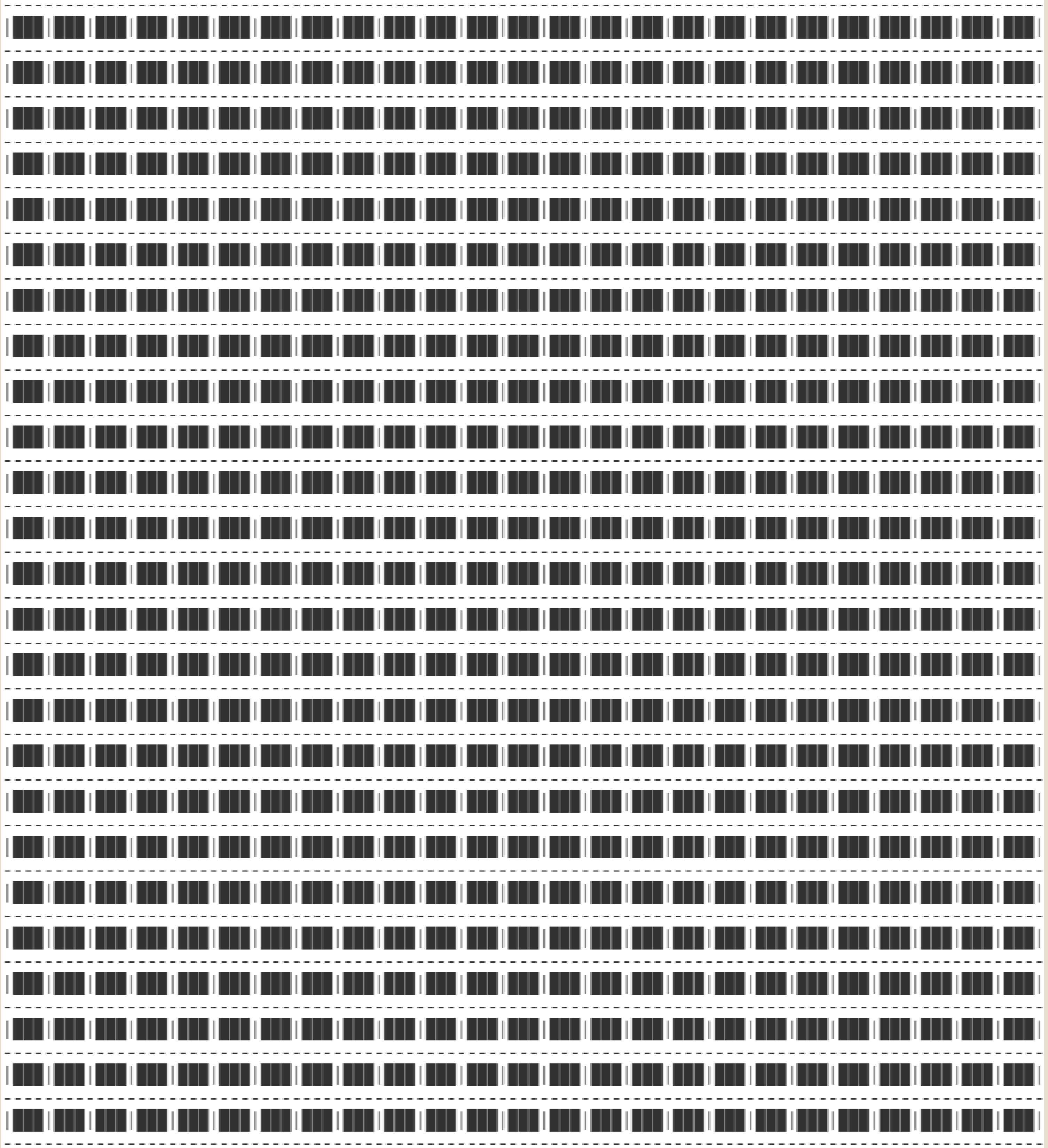


```
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   | x |   | x |   | x |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   | x |   |   |   |   |   |   | x |   |   |   |   | x |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   | x |   |   |   |   |   |   |   | x |   | x |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | x |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | x |   |   |   |   |   |   | x |   | x |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | x |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   |   | x |
-----------------------------------------------------------------------------------------------------
|   |   | x |   |   | x |   |   |   | x |   |   |   | x | x |   |   |   |   | x |   |   |   | x |   |
-----------------------------------------------------------------------------------------------------
| x |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   |   |   |   |   |   | x |   |
-----------------------------------------------------------------------------------------------------
| x |   | x |   |   |   |   |   |   | x |   | x |   |   |   |   |   | x | x |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   | x |   |   |   |   |   | x |   |   |   |   |   |   |   |   |   | x |   |   |   | x |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   | x |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   | x |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   | x |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   | x |   | x |   |   |   |   |   | x |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x | x |   |   |   |   |   |   | x | x |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   | x |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   | x |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   | x |   |   |   |   |   |   | x |   |   | x |   |   |   | x |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   | x |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   | x |   | x |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   |   |   |   | x |   |   |   |   |
-----------------------------------------------------------------------------------------------------
| x |   |   |   |   |   |   |   | x |   |   |   |   |   |   |   |   |   |   |   |   | x |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   | x |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
| x |   |   | x | x |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   | x |   |   | x | x |   |   |   | x |   |   |   |   |   | x |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
```

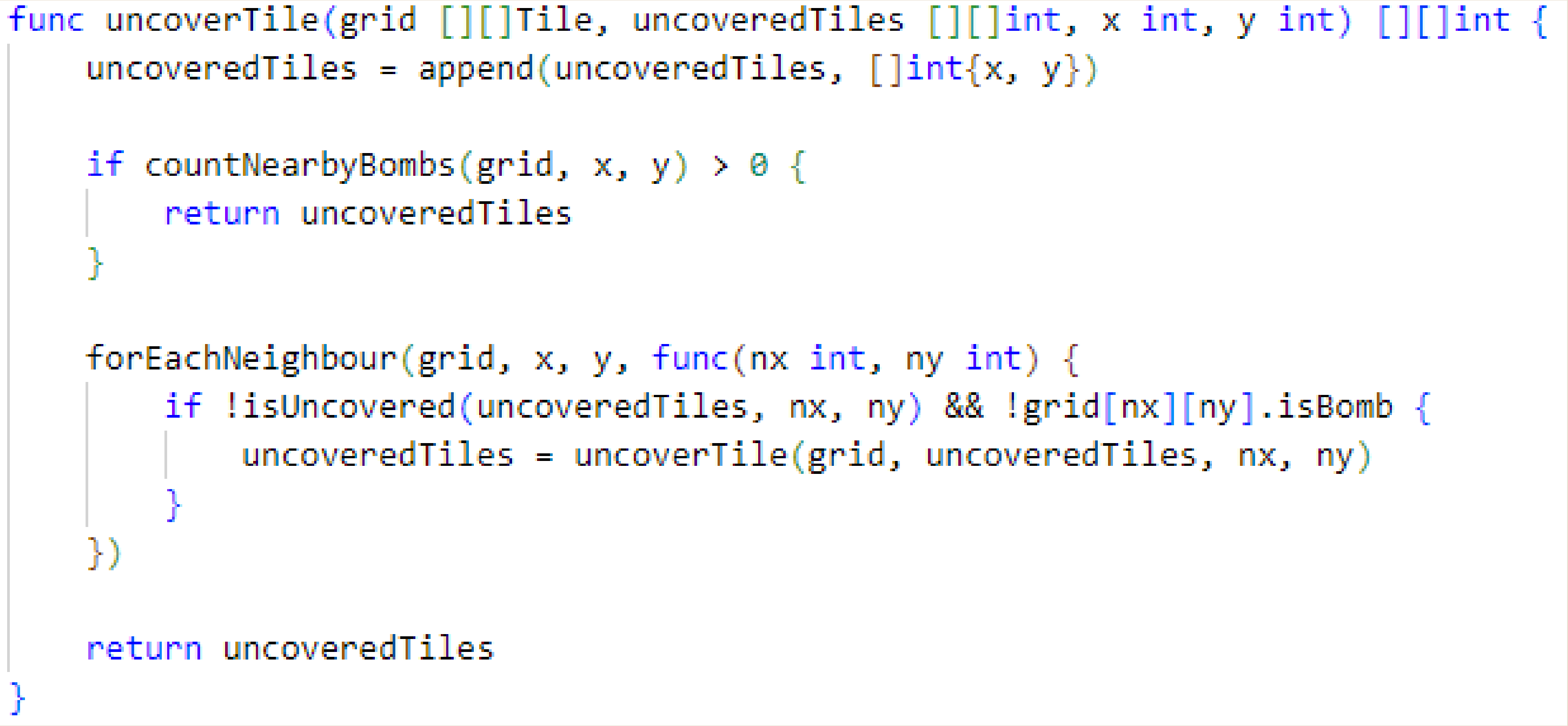
RÉSOLUTION DE LA GRILLE



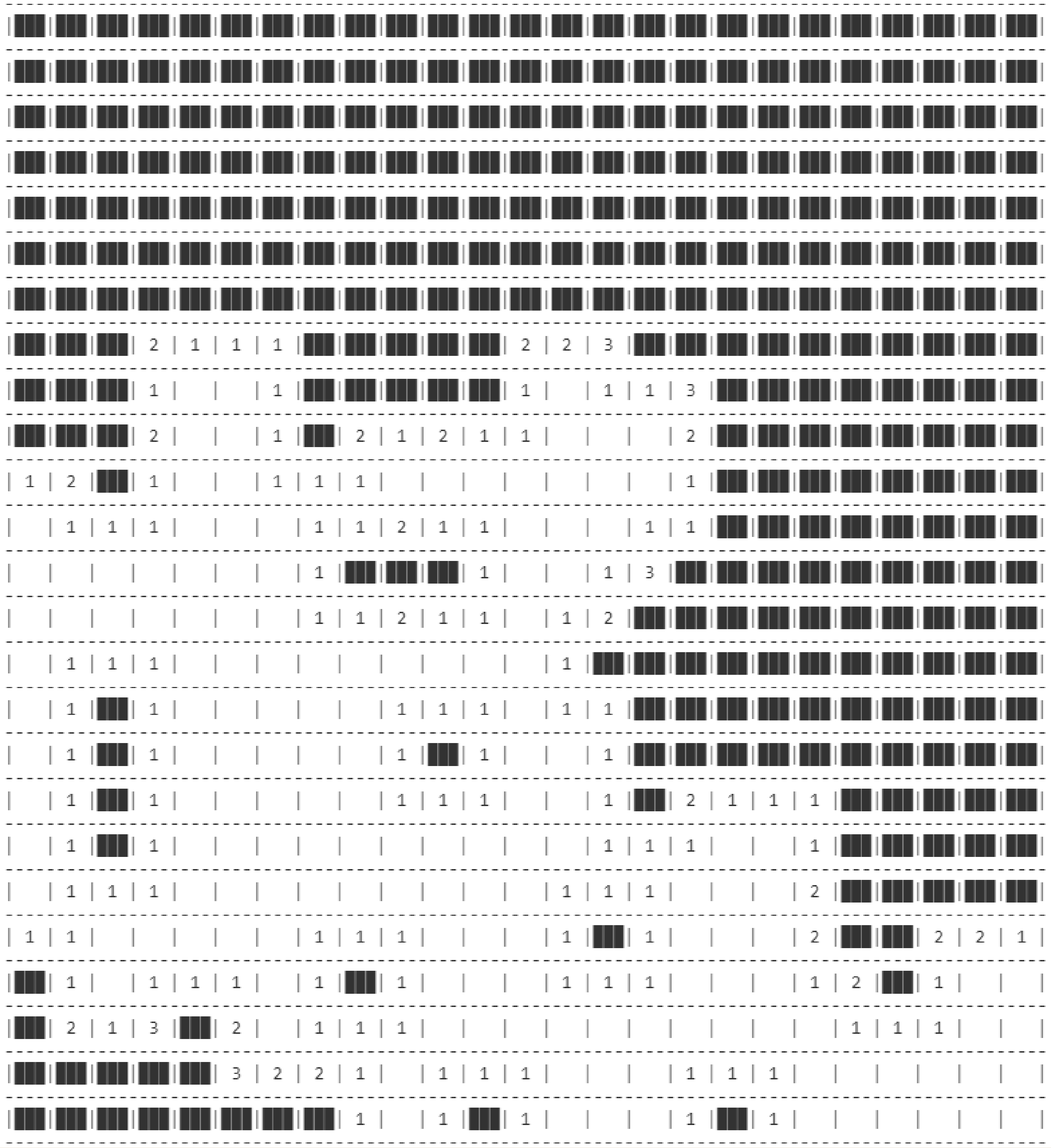
```go
func solve(grid [][]Tile, bombCount int) {
    gridSize := len(grid)
    var uncoveredTiles [][]int
    var flaggedTiles [][]int

    hasFailed := false
    x := RandomGenerator.Intn(gridSize - 1)
    y := RandomGenerator.Intn(gridSize - 1)

    for {
        if grid[x][y].isBomb {
            hasFailed = true
            break
        }
```



RÉSOLUTION DE LA GRILLE

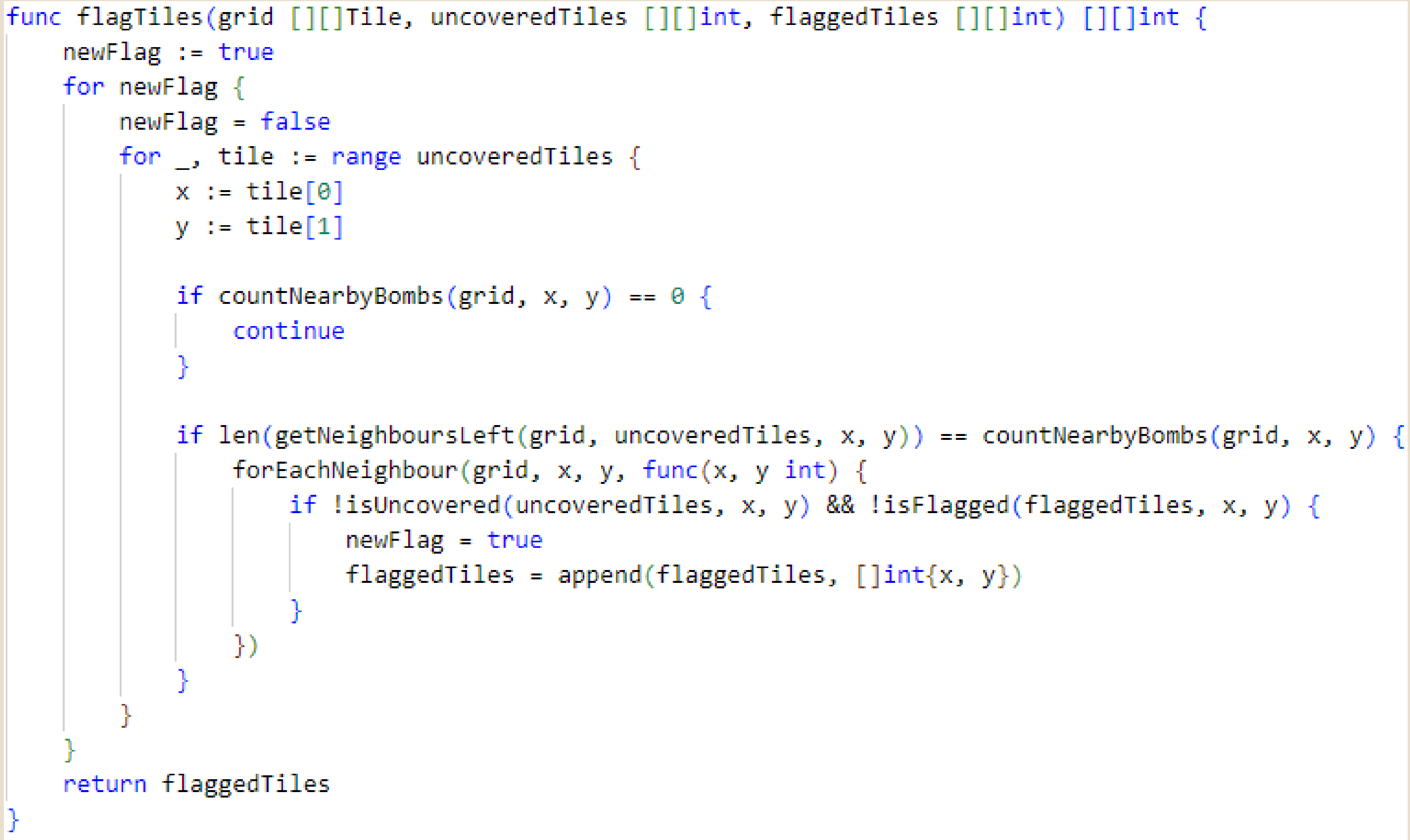


```go
func uncoverTile(grid [][]Tile, uncoveredTiles [][]int, x int, y int) [][]int {
    uncoveredTiles = append(uncoveredTiles, []int{x, y})

    if countNearbyBombs(grid, x, y) > 0 {
        return uncoveredTiles
    }

    forEachNeighbour(grid, x, y, func(nx int, ny int) {
        if !isUncovered(uncoveredTiles, nx, ny) && !grid[nx][ny].isBomb {
            uncoveredTiles = uncoverTile(grid, uncoveredTiles, nx, ny)
        }
    })

    return uncoveredTiles
}
```

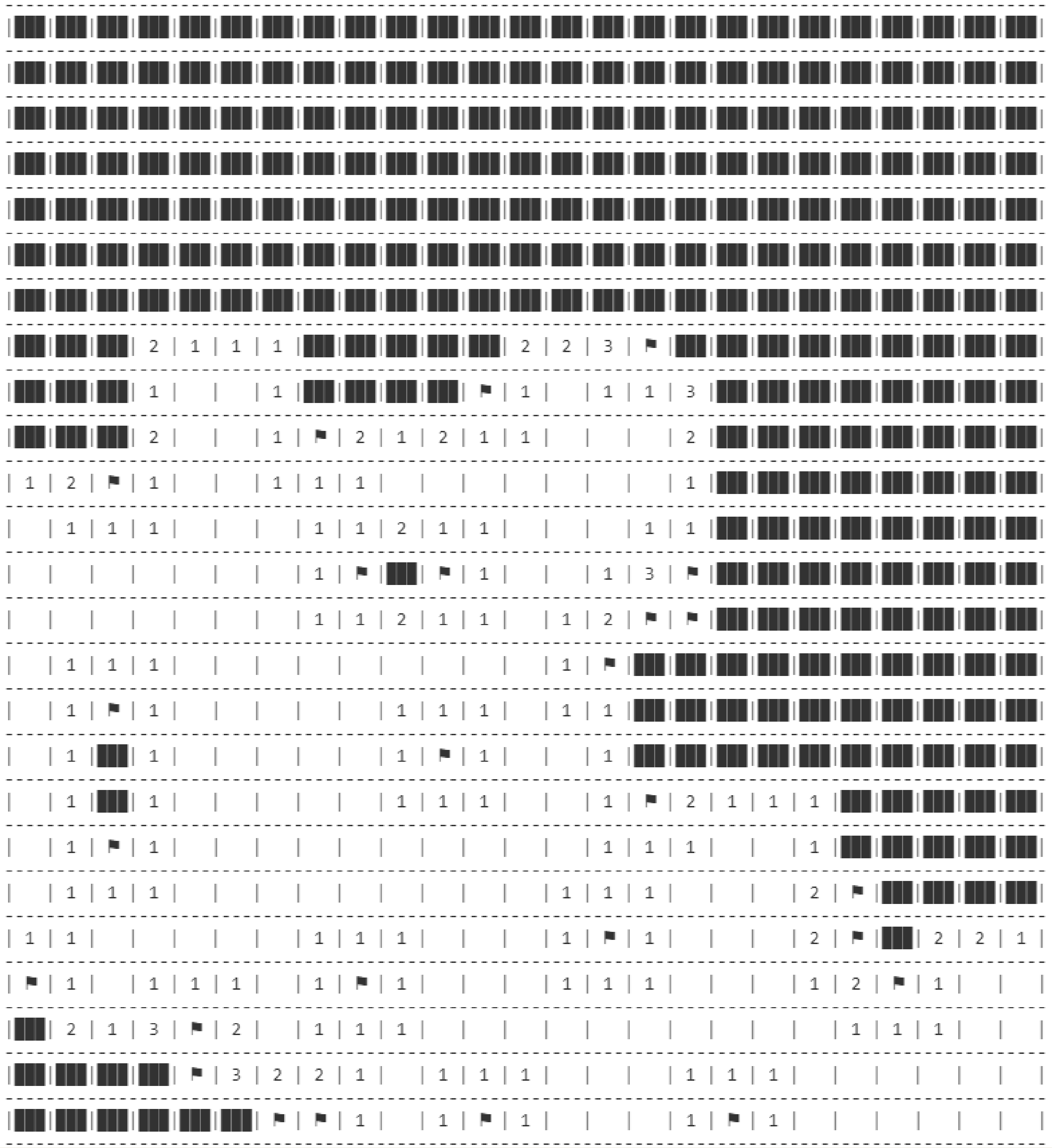


```
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 2 | 1 | 1 | 1 |███|███|███|███|███| 2 | 2 | 3 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 1 |   |   | 1 |███|███|███|███|███| 1 |   | 1 | 1 | 3 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 2 |   |   | 1 |███| 2 | 1 | 2 | 1 | 1 |   |   |   | 2 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
| 1 | 2 |███| 1 |   |   | 1 | 1 | 1 |   |   |   |   |   |   |   | 1 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   | 1 | 1 | 2 | 1 | 1 |   |   |   | 1 | 1 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 |███|███|███| 1 |   |   | 1 | 3 |███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | 1 | 2 | 1 | 1 |   | 1 | 2 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 |███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 | 1 | 1 |   | 1 | 1 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 |███| 1 |   |   | 1 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 |███| 2 | 1 | 1 | 1 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 2 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
| 1 | 1 |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 1 |███| 1 |   |   |   | 2 |███|███| 2 | 2 | 1 |
-----------------------------------------------------------------------------------------------------
|███| 1 |   | 1 | 1 | 1 |   | 1 |███| 1 |   |   |   | 1 | 1 | 1 |   |   |   | 1 | 2 |███| 1 |   |   |
-----------------------------------------------------------------------------------------------------
|███| 2 | 1 | 3 |███| 2 |   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███| 3 | 2 | 2 | 1 |   | 1 | 1 | 1 |   |   | 1 | 1 | 1 |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███| 1 |   | 1 |███| 1 |   |   | 1 |███| 1 |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
```

RÉSOLUTION DE LA GRILLE

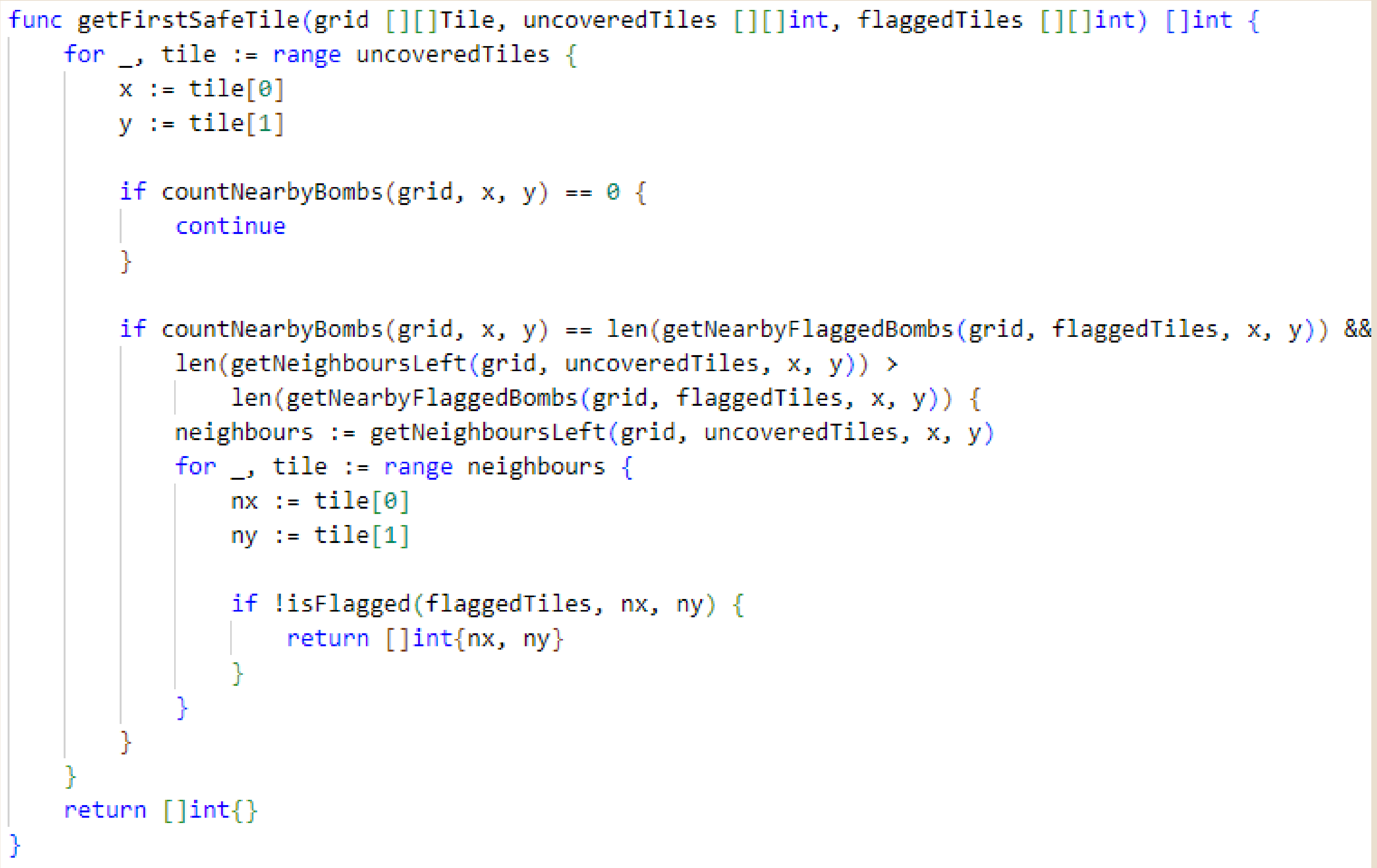


```go
func flagTiles(grid [][]Tile, uncoveredTiles [][]int, flaggedTiles [][]int) [][]int {
    newFlag := true
    for newFlag {
        newFlag = false
        for _, tile := range uncoveredTiles {
            x := tile[0]
            y := tile[1]

            if countNearbyBombs(grid, x, y) == 0 {
                continue
            }

            if len(getNeighboursLeft(grid, uncoveredTiles, x, y)) == countNearbyBombs(grid, x, y) {
                forEachNeighbour(grid, x, y, func(x, y int) {
                    if !isUncovered(uncoveredTiles, x, y) && !isFlagged(flaggedTiles, x, y) {
                        newFlag = true
                        flaggedTiles = append(flaggedTiles, []int{x, y})
                    }
                })
            }
        }
    }
    return flaggedTiles
}
```

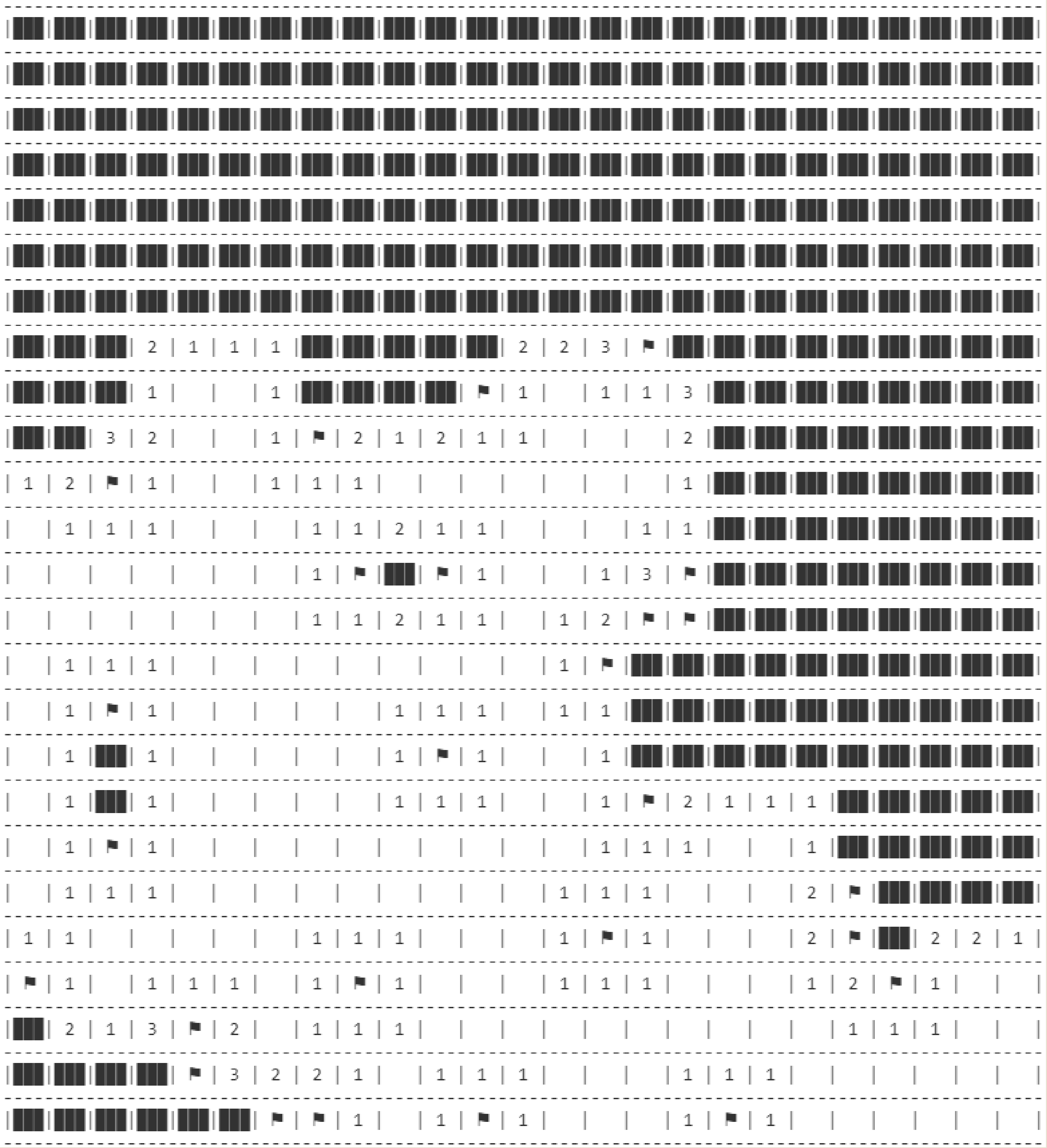


```
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 2 | 1 | 1 | 1 |███|███|███|███|███| 2 | 2 | 3 | ⚑ |███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 1 |   |   | 1 |███|███|███|███| ⚑ | 1 |   | 1 | 1 | 3 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 2 |   |   | 1 | ⚑ | 2 | 1 | 2 | 1 | 1 |   |   |   | 2 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
| 1 | 2 | ⚑ | 1 |   |   | 1 | 1 | 1 |   |   |   |   |   |   |   | 1 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   | 1 | 1 | 2 | 1 | 1 |   |   |   | 1 | 1 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | ⚑ |███| ⚑ | 1 |   |   | 1 | 3 | ⚑ |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | 1 | 2 | 1 | 1 |   | 1 | 2 | ⚑ | ⚑ |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | ⚑ |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 1 |   |   |   |   |   | 1 | 1 | 1 |   | 1 | 1 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 | ⚑ | 1 |   |   | 1 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 | ⚑ | 2 | 1 | 1 | 1 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 2 | ⚑ |███|███|███|███|
-----------------------------------------------------------------------------------------------------
| 1 | 1 |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 1 | ⚑ | 1 |   |   |   | 2 | ⚑ |███| 2 | 2 | 1 |
-----------------------------------------------------------------------------------------------------
| ⚑ | 1 |   | 1 | 1 | 1 |   | 1 | ⚑ | 1 |   |   |   | 1 | 1 | 1 |   |   |   | 1 | 2 | ⚑ | 1 |   |   |
-----------------------------------------------------------------------------------------------------
|███| 2 | 1 | 3 | ⚑ | 2 |   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |
-----------------------------------------------------------------------------------------------------
|███|███|███|███| ⚑ | 3 | 2 | 2 | 1 |   | 1 | 1 | 1 |   |   |   | 1 | 1 | 1 |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███| ⚑ | ⚑ | 1 |   | 1 | ⚑ | 1 |   |   |   | 1 | ⚑ | 1 |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
```

RÉSOLUTION DE LA GRILLE



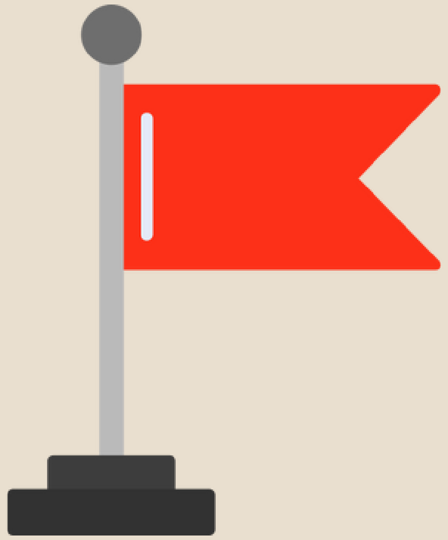
```go
func getFirstSafeTile(grid [][]Tile, uncoveredTiles [][]int, flaggedTiles [][]int) []int {
    for _, tile := range uncoveredTiles {
        x := tile[0]
        y := tile[1]

        if countNearbyBombs(grid, x, y) == 0 {
            continue
        }

        if countNearbyBombs(grid, x, y) == len(getNearbyFlaggedBombs(grid, flaggedTiles, x, y)) &&
            len(getNeighboursLeft(grid, uncoveredTiles, x, y)) >
                len(getNearbyFlaggedBombs(grid, flaggedTiles, x, y)) {
            neighbours := getNeighboursLeft(grid, uncoveredTiles, x, y)
            for _, tile := range neighbours {
                nx := tile[0]
                ny := tile[1]

                if !isFlagged(flaggedTiles, nx, ny) {
                    return []int{nx, ny}
                }
            }
        }
    }
    return []int{}
}
```

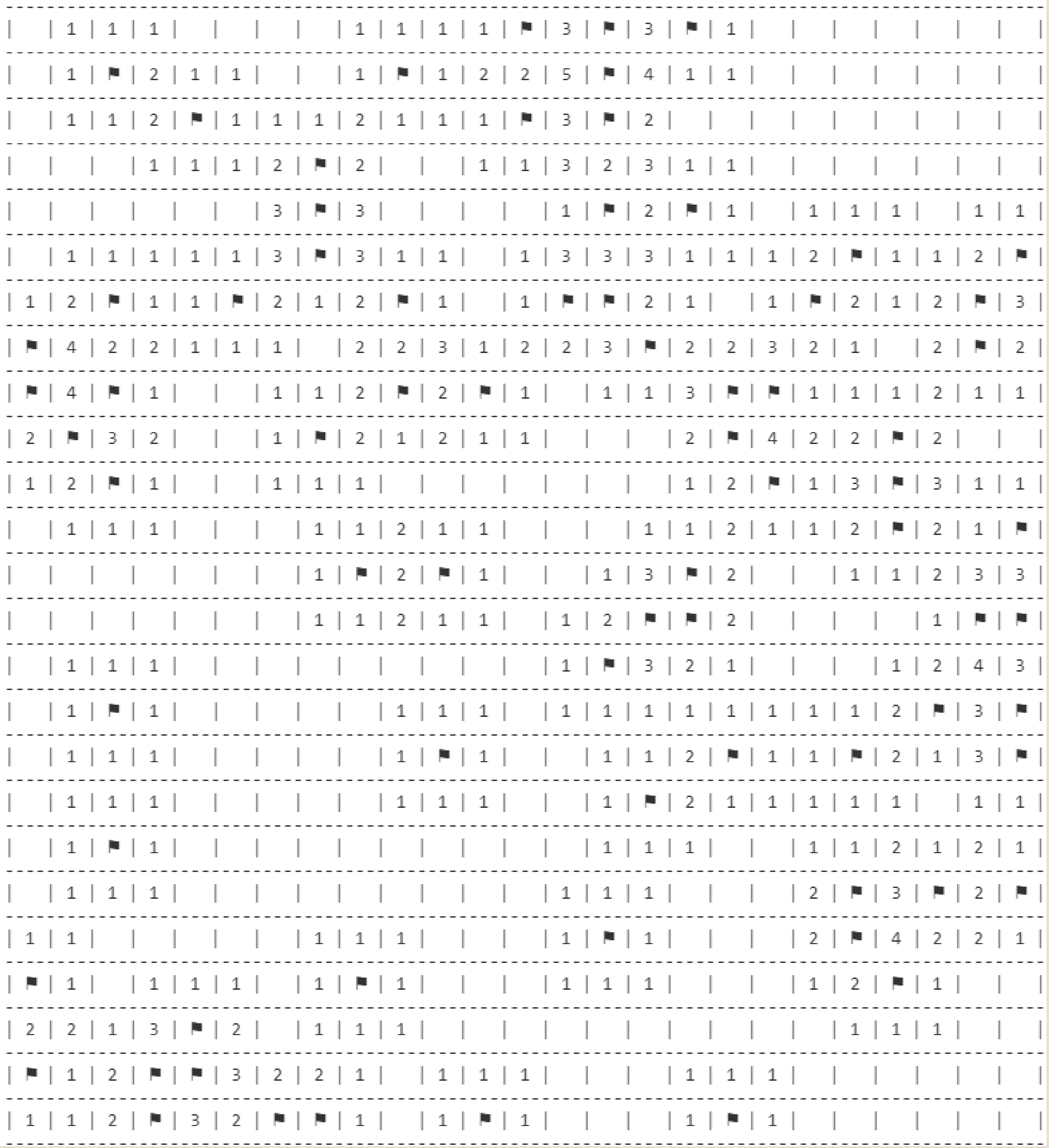


```
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 2 | 1 | 1 | 1 |███|███|███|███|███| 2 | 2 | 3 | ⚑ |███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███|███| 1 |   |   | 1 |███|███|███|███| ⚑ | 1 |   | 1 | 1 | 3 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|███|███| 3 | 2 |   |   | 1 | ⚑ | 2 | 1 | 2 | 1 | 1 |   |   |   | 2 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
| 1 | 2 | ⚑ | 1 |   |   | 1 | 1 | 1 |   |   |   |   |   |   |   | 1 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   | 1 | 1 | 2 | 1 | 1 |   |   |   | 1 | 1 |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | ⚑ |███| ⚑ | 1 |   |   | 1 | 3 | ⚑ |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | 1 | 2 | 1 | 1 |   | 1 | 2 | ⚑ | ⚑ |███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | ⚑ |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 1 |   |   |   |   |   | 1 | 1 | 1 |   | 1 | 1 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 | ⚑ | 1 |   |   | 1 |███|███|███|███|███|███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 |███| 1 |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 | ⚑ | 2 | 1 | 1 | 1 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 |███|███|███|███|███|
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 2 | ⚑ |███|███|███|███|
-----------------------------------------------------------------------------------------------------
| 1 | 1 |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 1 | ⚑ | 1 |   |   |   | 2 | ⚑ |███| 2 | 2 | 1 |
-----------------------------------------------------------------------------------------------------
| ⚑ | 1 |   | 1 | 1 | 1 |   | 1 | ⚑ | 1 |   |   |   | 1 | 1 | 1 |   |   |   | 1 | 2 | ⚑ | 1 |   |   |
-----------------------------------------------------------------------------------------------------
|███| 2 | 1 | 3 | ⚑ | 2 |   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |
-----------------------------------------------------------------------------------------------------
|███|███|███|███| ⚑ | 3 | 2 | 2 | 1 |   | 1 | 1 | 1 |   |   |   | 1 | 1 | 1 |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|███|███|███|███|███|███| ⚑ | ⚑ | 1 |   | 1 | ⚑ | 1 |   |   |   | 1 | ⚑ | 1 |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
```

RÉSOLUTION DE LA GRILLE

BIEN JOUÉ LA ZONE !





```
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   | 1 | 1 | 1 | 1 | ⚑ | 3 | ⚑ | 3 | ⚑ | 1 |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 2 | 1 | 1 |   |   | 1 | ⚑ | 1 | 2 | 2 | 5 | ⚑ | 4 | 1 | 1 |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 2 | ⚑ | 1 | 1 | 1 | 2 | 1 | 1 | 1 | ⚑ | 3 | ⚑ | 2 |   |   |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   | 1 | 1 | 1 | 2 | ⚑ | 2 |   |   | 1 | 1 | 3 | 2 | 3 | 1 | 1 |   |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   | 3 | ⚑ | 3 |   |   |   |   | 1 | ⚑ | 2 | ⚑ | 1 |   | 1 | 1 | 1 |   | 1 | 1 |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 | 1 | 1 | 3 | ⚑ | 3 | 1 | 1 |   | 1 | 3 | 3 | 3 | 1 | 1 | 1 | 2 | ⚑ | 1 | 1 | 2 | ⚑ |
-----------------------------------------------------------------------------------------------------
| 1 | 2 | ⚑ | 1 | 1 | ⚑ | 2 | 1 | 2 | ⚑ | 1 |   | 1 | ⚑ | ⚑ | 2 | 1 |   | 1 | ⚑ | 2 | 1 | 2 | ⚑ | 3 |
-----------------------------------------------------------------------------------------------------
| ⚑ | 4 | 2 | 2 | 1 | 1 | 1 |   | 2 | 2 | 3 | 1 | 2 | 2 | 3 | ⚑ | 2 | 2 | 3 | 2 | 1 |   | 2 | ⚑ | 2 |
-----------------------------------------------------------------------------------------------------
| ⚑ | 4 | ⚑ | 1 |   |   | 1 | 1 | 2 | ⚑ | 2 | ⚑ | 1 |   | 1 | 1 | 3 | ⚑ | ⚑ | 1 | 1 | 1 | 2 | 1 | 1 |
-----------------------------------------------------------------------------------------------------
| 2 | ⚑ | 3 | 2 |   |   | 1 | ⚑ | 2 | 1 | 2 | 1 | 1 |   |   |   | 2 | ⚑ | 4 | 2 | 2 | ⚑ | 2 |   |   |
-----------------------------------------------------------------------------------------------------
| 1 | 2 | ⚑ | 1 |   |   | 1 | 1 | 1 |   |   |   |   |   |   |   | 1 | 2 | ⚑ | 1 | 3 | ⚑ | 3 | 1 | 1 |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   | 1 | 1 | 2 | 1 | 1 |   |   |   | 1 | 1 | 2 | 1 | 1 | 2 | ⚑ | 2 | 1 | ⚑ |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | ⚑ | 2 | ⚑ | 1 |   |   | 1 | 3 | ⚑ | 2 |   |   | 1 | 1 | 2 | 3 | 3 |
-----------------------------------------------------------------------------------------------------
|   |   |   |   |   |   |   | 1 | 1 | 2 | 1 | 1 |   | 1 | 2 | ⚑ | ⚑ | 2 |   |   |   |   | 1 | ⚑ | ⚑ |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | ⚑ | 3 | 2 | 1 |   |   |   | 1 | 2 | 4 | 3 |
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 1 |   |   |   |   |   | 1 | 1 | 1 |   | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | ⚑ | 3 | ⚑ |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   | 1 | ⚑ | 1 |   |   | 1 | 1 | 2 | ⚑ | 1 | 1 | ⚑ | 2 | 1 | 3 | ⚑ |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 | ⚑ | 2 | 1 | 1 | 1 | 1 | 1 |   | 1 | 1 |
-----------------------------------------------------------------------------------------------------
|   | 1 | ⚑ | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   | 1 | 1 | 2 | 1 | 2 | 1 |
-----------------------------------------------------------------------------------------------------
|   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 2 | ⚑ | 3 | ⚑ | 2 | ⚑ |
-----------------------------------------------------------------------------------------------------
| 1 | 1 |   |   |   |   |   | 1 | 1 | 1 |   |   |   | 1 | ⚑ | 1 |   |   |   | 2 | ⚑ | 4 | 2 | 2 | 1 |
-----------------------------------------------------------------------------------------------------
| ⚑ | 1 |   | 1 | 1 | 1 |   | 1 | ⚑ | 1 |   |   |   | 1 | 1 | 1 |   |   |   | 1 | 2 | ⚑ | 1 |   |   |
-----------------------------------------------------------------------------------------------------
| 2 | 2 | 1 | 3 | ⚑ | 2 |   | 1 | 1 | 1 |   |   |   |   |   |   |   |   |   |   | 1 | 1 | 1 |   |   |
-----------------------------------------------------------------------------------------------------
| ⚑ | 1 | 2 | ⚑ | ⚑ | 3 | 2 | 2 | 1 |   | 1 | 1 | 1 |   |   |   | 1 | 1 | 1 |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
| 1 | 1 | 2 | ⚑ | 3 | 2 | ⚑ | ⚑ | 1 |   | 1 | ⚑ | 1 |   |   |   | 1 | ⚑ | 1 |   |   |   |   |   |   |
-----------------------------------------------------------------------------------------------------
```

NAÏVE

PERF GRID

Vitesse : 1475 ns

Mémoire : 1.4 KB

Allocations Mémoire : 26

PERF SOLVE

Vitesse : 29.7 s

Mémoire : 840.21 KB

Allocations Mémoire : 350 millions

OPTIMISATION 1

CALCUL DES INDICES BOMB

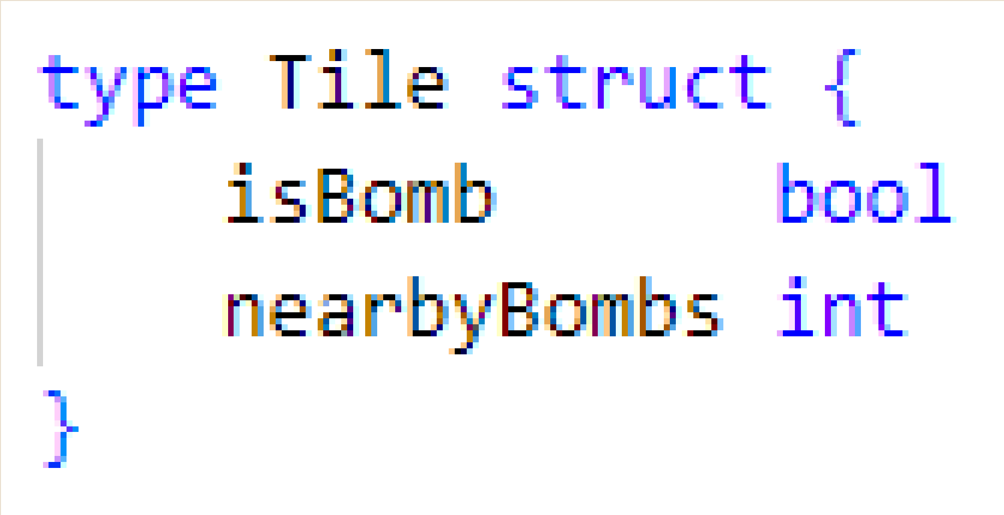
Avant : à chaque opération

- *Découvrement*
- *Marquage de bombes*
- *Détection de cases sûres*

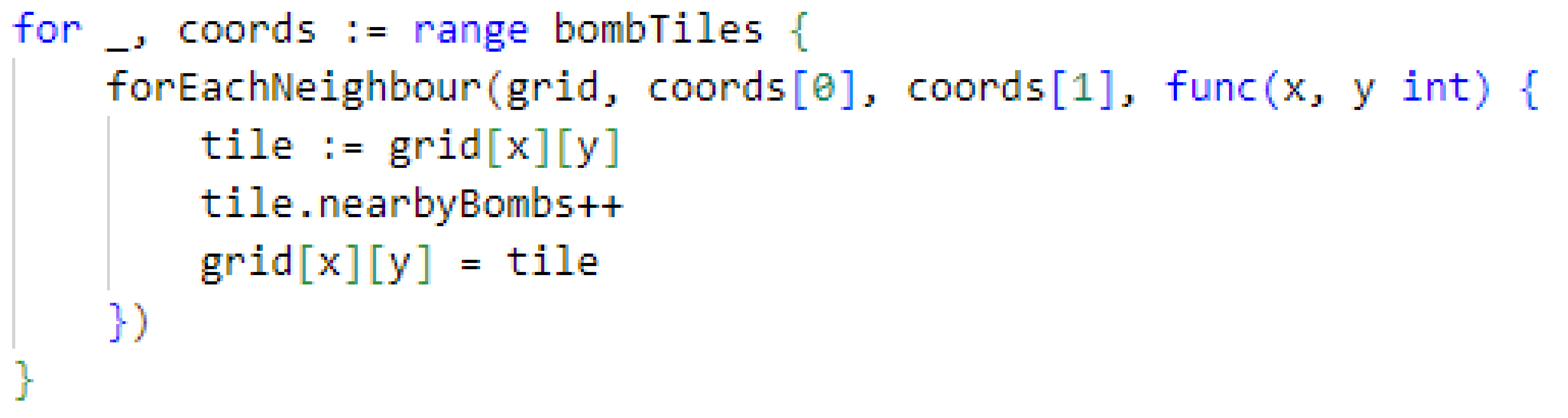
Après :

- *Calcul à la génération*
- *Stockage dans l'objet Tile*

OPTIMISATION 1



```go
type Tile struct {
    isBomb      bool
    nearbyBombs int
}
```



```go
for _, coords := range bombTiles {
    forEachNeighbour(grid, coords[0], coords[1], func(x, y int) {
        tile := grid[x][y]
        tile.nearbyBombs++
        grid[x][y] = tile
    })
}
```

OPTIMISATION 1

PERF GRID

Vitesse : 7 733 ns

Mémoire : 19.7 KB

Allocations Mémoire : 109

PERF SOLVE

Vitesse : 23.2 s

Mémoire : 840.21 KB

Allocations Mémoire : 350 millions

OPTIMISATION 2

PLUS DE STOCKAGE DANS TILE

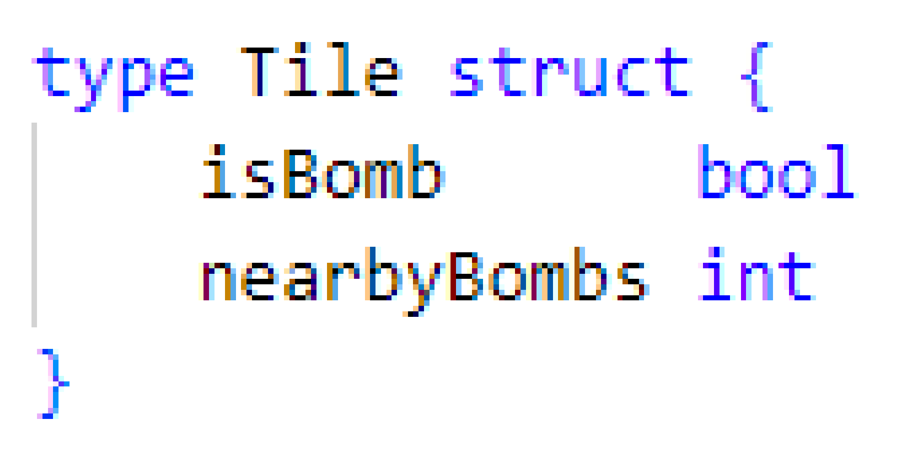
Avant :

- *Tableau cases découvertes*
- *Tableau cases marquées*

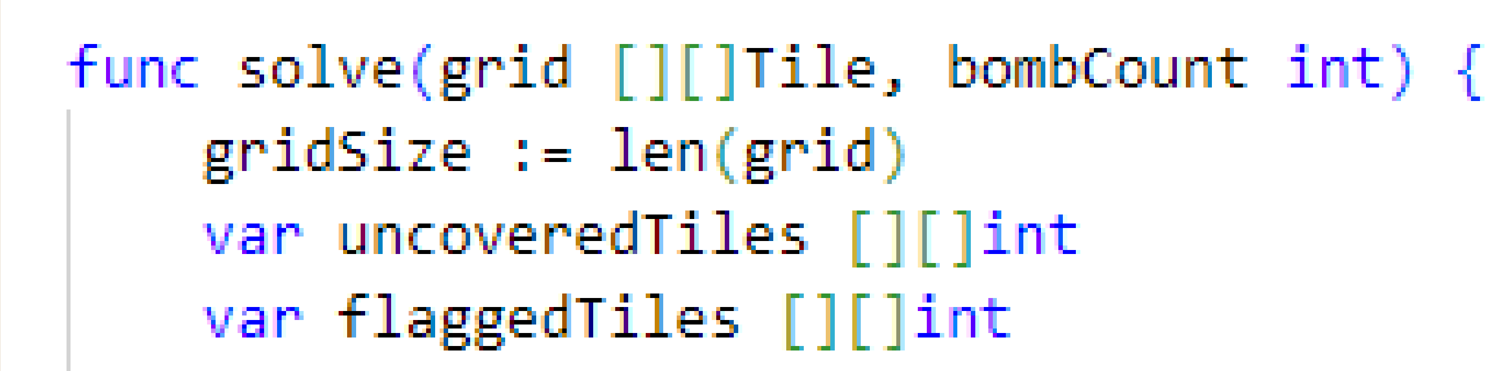
Après :

- *Paramètres isUncovered, isFlagged, x et y dans Tile*
- *Plus de comparaisons de tableaux*

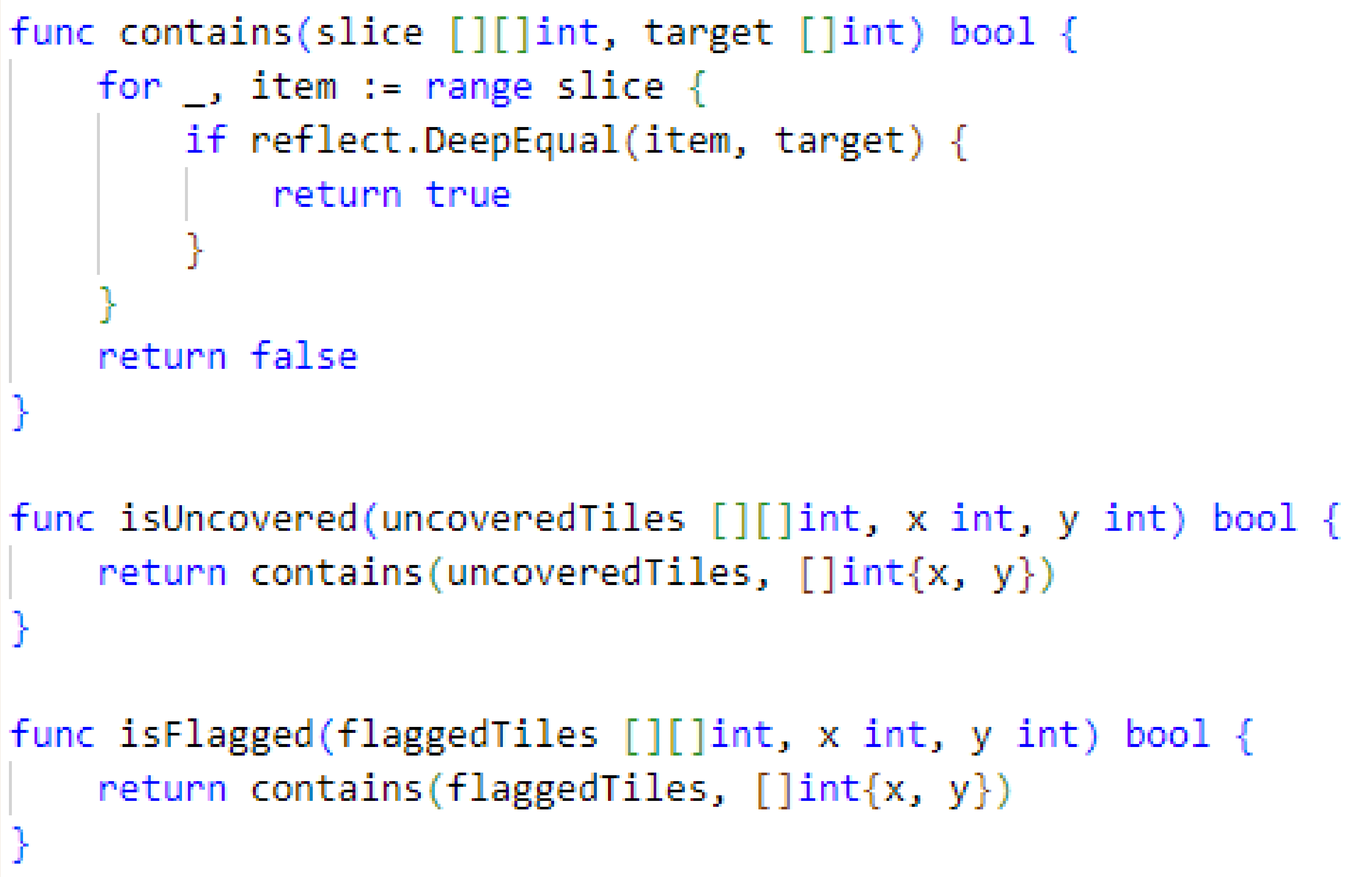
AVANT



```go
type Tile struct {
    isBomb      bool
    nearbyBombs int
}
```

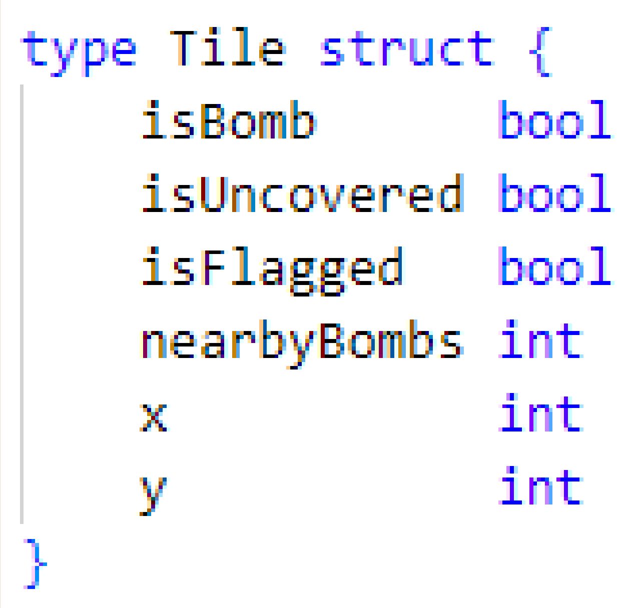


```go
func solve(grid [][]Tile, bombCount int) {
    gridSize := len(grid)
    var uncoveredTiles [][]int
    var flaggedTiles [][]int
```

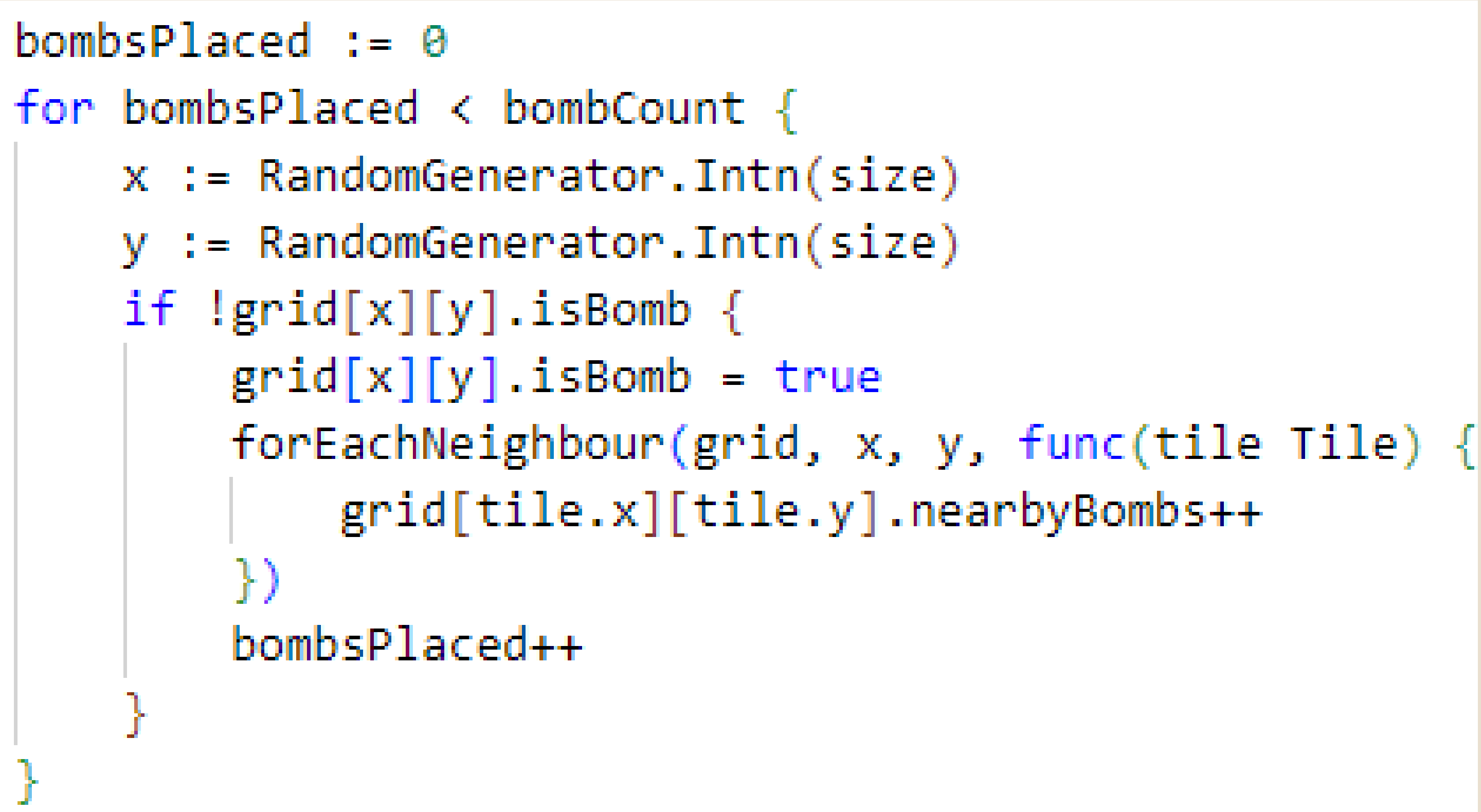


```go
func contains(slice [][]int, target []int) bool {
    for _, item := range slice {
        if reflect.DeepEqual(item, target) {
            return true
        }
    }
    return false
}

func isUncovered(uncoveredTiles [][]int, x int, y int) bool {
    return contains(uncoveredTiles, []int{x, y})
}

func isFlagged(flaggedTiles [][]int, x int, y int) bool {
    return contains(flaggedTiles, []int{x, y})
}
```

APRES



```
type Tile struct {
	isBomb      bool
	isUncovered bool
	isFlagged   bool
	nearbyBombs int
	x           int
	y           int
}
```



```
bombsPlaced := 0
for bombsPlaced < bombCount {
	x := RandomGenerator.Intn(size)
	y := RandomGenerator.Intn(size)
	if !grid[x][y].isBomb {
		grid[x][y].isBomb = true
		forEachNeighbour(grid, x, y, func(tile Tile) {
			grid[tile.x][tile.y].nearbyBombs++
		})
		bombsPlaced++
	}
}
```

OPTIMISATION 2

PERF GRID

Vitesse : 8 748 ns

Mémoire : 23 KB

Allocations Mémoire : 26

PERF SOLVE

Vitesse : 3.3 s

Mémoire : 410.8 KB

Allocations Mémoire : 75 120

OPTIMISATION 3

UTILISATION DE GOROUTINES

Avant :

- *Aucune Goroutine utilisé*

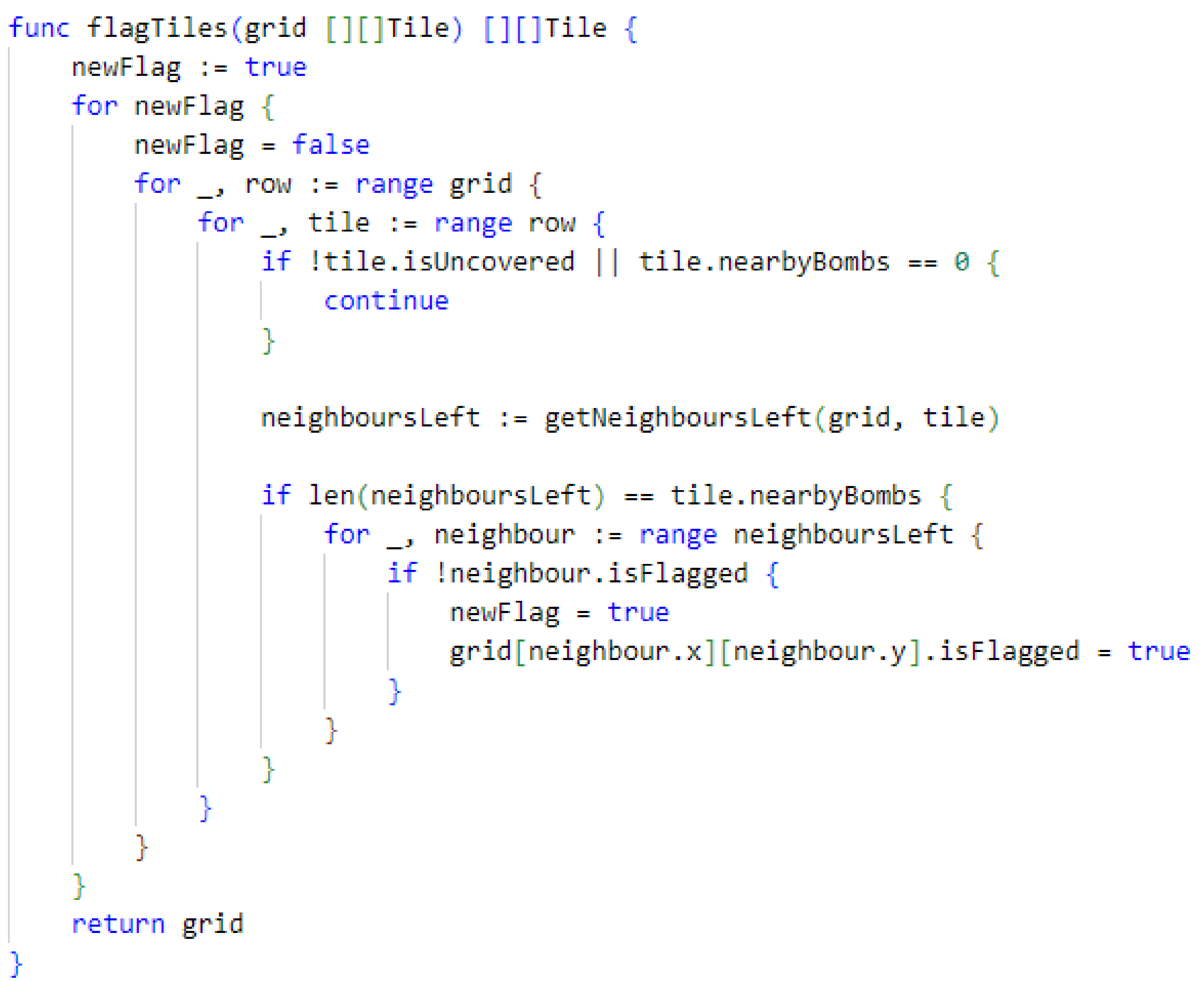
Après :

- *Utilisation de Goroutine pour le marquage de bombes*

Limite :

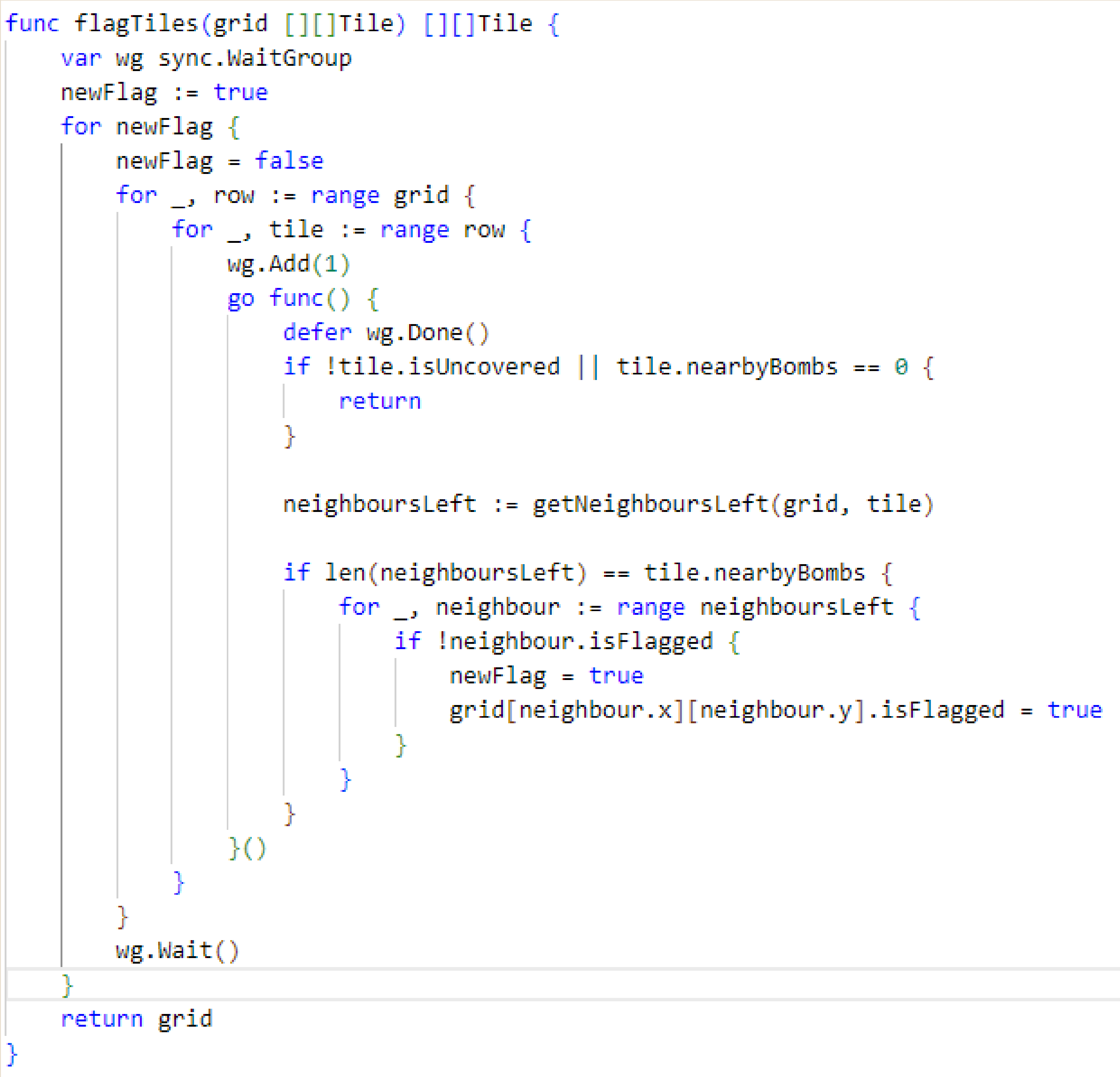
- *Dévoilement des cases*

AVANT



```go
func flagTiles(grid [][]Tile) [][]Tile {
    newFlag := true
    for newFlag {
        newFlag = false
        for _, row := range grid {
            for _, tile := range row {
                if !tile.isUncovered || tile.nearbyBombs == 0 {
                    continue
                }

                neighboursLeft := getNeighboursLeft(grid, tile)

                if len(neighboursLeft) == tile.nearbyBombs {
                    for _, neighbour := range neighboursLeft {
                        if !neighbour.isFlagged {
                            newFlag = true
                            grid[neighbour.x][neighbour.y].isFlagged = true
                        }
                    }
                }
            }
        }
    }
    return grid
}
```

APRES



```go
func flagTiles(grid [][]Tile) [][]Tile {
    var wg sync.WaitGroup
    newFlag := true
    for newFlag {
        newFlag = false
        for _, row := range grid {
            for _, tile := range row {
                wg.Add(1)
                go func() {
                    defer wg.Done()
                    if !tile.isUncovered || tile.nearbyBombs == 0 {
                        return
                    }

                    neighboursLeft := getNeighboursLeft(grid, tile)

                    if len(neighboursLeft) == tile.nearbyBombs {
                        for _, neighbour := range neighboursLeft {
                            if !neighbour.isFlagged {
                                newFlag = true
                                grid[neighbour.x][neighbour.y].isFlagged = true
                            }
                        }
                    }
                }()
            }
        }
        wg.Wait()
    }
    return grid
}
```

OPTIMISATION 3

PERF GRID

Vitesse : 8 693 ns

Mémoire : 23 KB

Allocations Mémoire : 26

PERF SOLVE

Vitesse : 2.7 s

Mémoire : 109.1 KB

Allocations Mémoire : 160 346

CONCLUSION

Génération de la grille					
	Solution simple	Optimisation 1	Optimisation 2	Optimisation 3	Delta
Vitesse (ns)	1475	7 733	8 748	8 693	-7,218
Mémoire (KB)	1.4	19.7	23	23	-21.6
Allocation mémoires	26	109	26	26	0

Résolution de la grille					
	Solution simple	Optimisation 1	Optimisation 2	Optimisation 3	Delta
Vitesse (s)	29.7	23.2	3.3	2.7	-27
Mémoire (KB)	840	840	410	109	-731
Allocation mémoires	350 M	350 M	75 120	160 346	-~350M