N-QUEENS

*Vocational School for Computer Programming and innovation*

A project by:

Team - KKB

Contents

[1. Our Team 2](#_Toc62325746)

[2. Main functions description 3](#_Toc62325747)

[3. Block diagram 5](#_Toc62325748)

[4. Used technology 6](#_Toc62325749)

[5. How the code works 6](#_Toc62325750)

# Our Team

##### Scrum trainer:

Hussein Abumelih - [hsabumelih18@codingburgas.bg](mailto:AFYusrefova18@codingburgas.bg)

##### Developers:

**Back end:**

Kaloqn Dinev - [kddinev18@codingburgas.bg](mailto:kddinev18@codingburgas.bg)

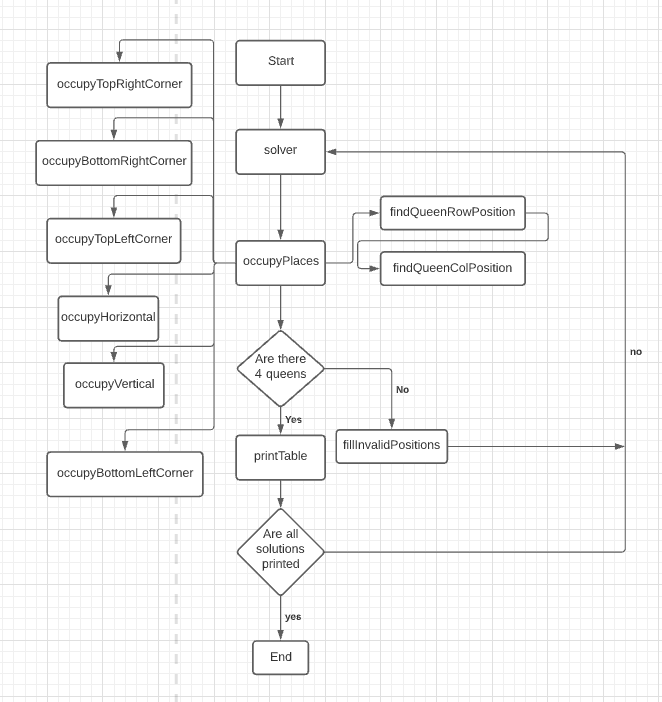
**Front end:**

Kaloyan Yordanov - [kyayordanov18@codingburgas.bg](mailto:kyayordanov18@codingburgas.bg)

# Main functions description

|  |  |  |
| --- | --- | --- |
| **Name and type** | **Purpose** | Arguments |
| int findQueenRowPosition | Finds on what position on the row is the queen | int col = 0,  int row = 0 |
| int findQueenColPosition | Finds on what position on the Column is the queen | int col = 0,  int row = 0 |
| void occupyTopRightCorner | Occupy positions from the queens to top right corner | int row, int col |
| void occupyTopLeftCorner | Occupy positions from the queens to top left corner | int row, int col |
| void occupyBottomRightCorner | Occupy positions from the queens to Bottom right corner | int row, int col |
| void occupyBottomLeftCorner | Occupy positions from the queens to Bottom Left corner | int row, int col |
| void occupyHorizontal | occupy the whole row, where is the queen | int col,  int i = 0 |
| void occupyVertical | occupy the whole column, where is the queen | int row,  int i = 0 |
| void occupyPlaces | decides what occupy finction to use |  |
| void printTable |  |  |
| int clearBoard |  | int fillerRow,  int fillerCol |
| void fillInvalidPositions |  |  |
| void solver | ocupy the first accessible position and check if all solutions are printed | int fillerRow = 0, int fillerCol = 0 |

# Block diagram



`

# Used technology

# How the code works

The program will set the queens in the first accessible position.

Then it will fill up the positions where the queen can strike.

Then Again it will place the next queen in the first accessible position.

And again, it will fill up the positions where the queen can strike.



In this case we can have only 3 queens. So, the board will clean up and the first position will be filled.

Now when we have come to the first solution, the board will clean up and the second position will be filled.



And this process will be repeated as many times as it takes to solve the problem.