|  |
| --- |
| quadSquad |

|  |
| --- |
|  |
| The Bletchley code breakers | |

|  |
| --- |
| 13.1.2021 г. |

Table of Contents

[1. Autors 2](#_Toc61353301)

[2. Summery 2](#_Toc61353302)

[2.1 Goals 2](#_Toc61353303)

[2.2 Main stages of realization 2](#_Toc61353304)

[2.3 Level of difficulty, main problems during realization 3](#_Toc61353305)

[2.4 Diagram 3](#_Toc61353306)

[3. Description of the functions used 4](#_Toc61353307)

# Autors

* Viktor Kavaldzhiev
* [vgkavaldzhiev18@codingburgas.bg](mailto:vgkavaldzhiev18@codingburgas.bg)
* SCRUM trainer
* Magdalena Omayska
* [miomayska18@codingburgas.bg](mailto:miomayska18@codingburgas.bg)
* Front-end developer
* Vasil Stanchev
* [VTStanchev18@codingburgas.bg](mailto:VTStanchev18@codingburgas.bg)
* C++ developer
* Quality engineer
* Yoana Stoyanova
* [yistoyanova18@codingburgas.bg](mailto:yistoyanova18@codingburgas.bg)
* Documentation
* Code checker

# 2. Summery

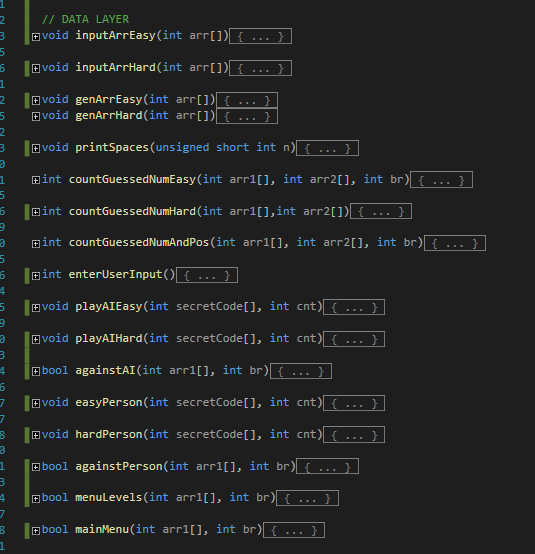
## 2.1 Goals

The goal of our project is to create a game that can be played by one or two players. Our team’s mission is to provide a program that allows the player to decipher the coordinates of the German battleships so that the allies can track them and destroy them. The game has two levels of difficulty for the people that want to challenge their minds.

## 2.2 Main stages of realization

First stage – planning. The first stage of the realization of our project was to create a meeting and discus the different roles and tasks that we had to do.

Second stage – realization. When the tasks and roles were clear we started to work on them. We began writing the project code and making the presentation and documentation that explain it in more detail.

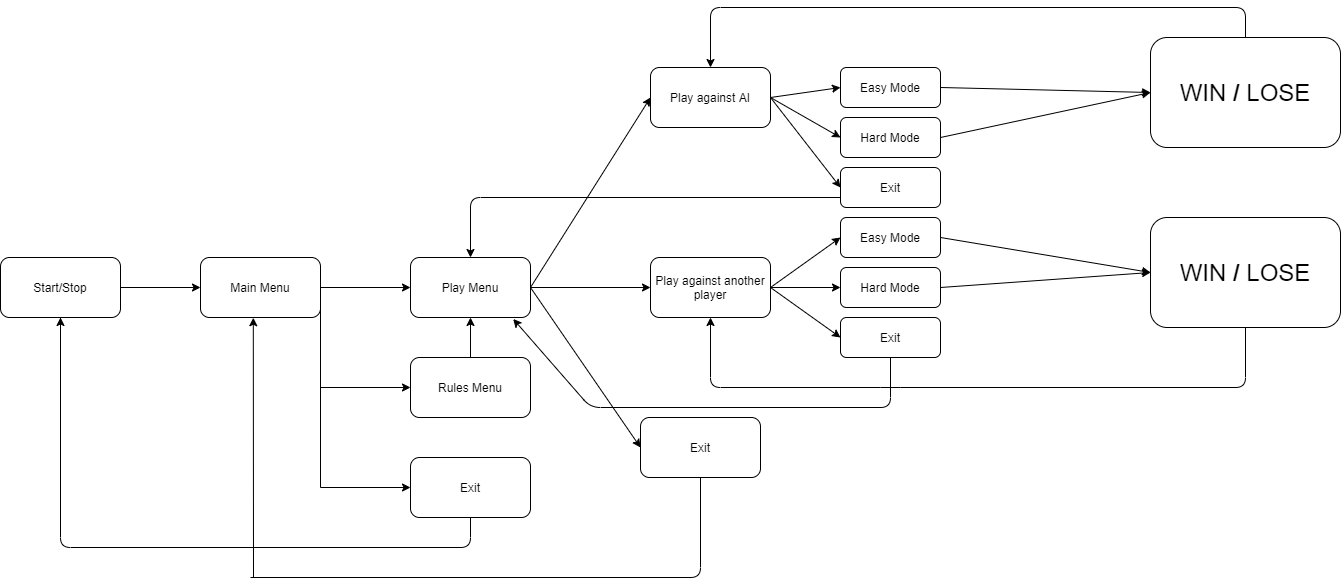


Third stage – presentation. The final stage is to present the finished product in front of the judges.

## 2.3 Level of difficulty, main problems during realization

Writing the code for the program was the most challenging part of completing our project. We had a few misunderstandings along the way about the functionality of various sections of the code. We had a few malfunctions with GitHub as well, but we were able to overcome everything and finish our tasks at the end.

## 2.4 Diagram



## 3. Description of the functions used

|  |  |  |  |
| --- | --- | --- | --- |
| Function name | Purpose | Arguments | Returned value |
| PRESENTATION LAYER | | | |
| printSpaces() | Prints spaces | unsigned short int n | void |
| initialGreeting() | Prints a short greeting when the program is started | none | void |
| rules() | Prints the rules of the game | none | void |
| showGoodbyeMessage() | Prints a short message when the program is ended | none | void |
| printWinMessage() | Prints a message when the user wins | none | void |
| printLoseMessage() | Prints a message when the user loses | int secretCode[] | void |
| printAIMenu() | Prints the menu for playing against a computer | none | void |
| printPersonMenu() | Prints the menu for playing against another player | none | void |
| printGameMenu() | Prints the menu with game options (either against a computer or a player) | none | void |
| printMainMenu() | Prints the main menu | none | void |
| DATA LAYER | | | |
| inputArrEasy() | Player inputs their guess in easy mode | int arr[] | void |
| inputArrHard() | Player inputs their guess in hard mode | int arr[] | void |
| genArrEasy() | Generates four-digit code without repeating digits | int arr[] | void |
| genArrHard() | Generates four-digit code with the chance of repeating digits | int arr[] | void |
| countGuessedNumEasy() | Counts guessed numbers in easy mode | int arr1[], int arr2[], int br | int |
| countGuessedNumHard() | Counts guessed numbers in hard mode | int arr1[],int arr2[] | int |
| countGuessedNumAndPos() | Counts guessed positions | int arr1[], int arr2[], int br | int |
| enterUserInput() | User enters numbers in array | none | void |
| playAIEasy() | Gameplay with computer in easy mode | int secretCode[], int cnt | void |
| playAIHard() | Gameplay against computer in hard mode | int secretCode[], int cnt | void |
| againstAI() | Menu for playing against a computer | int arr1[], int br | bool |
| easyPerson() | Gameplay with another player in easy mode | int secretCode[], int cnt | void |
| hardPerson() | Gameplay with another player in hard mode | int secretCode[], int cnt | void |
| againstPerson() | Menu for playing against another player | int arr1[], int br | bool |
| menuLevels() | Menu for choosing to play against a computer or a person | int arr1[], int br | bool |
| mainMenu() | Main menu | int arr1[], int br | bool |
| main() | Main function | none | int |