Vocational high school in computer programming

and innovation

**The Bletchley code breakers**

Team sossila.v2

|  |  |
| --- | --- |
| **Team Members** | **Role** |
| *Viktor Velizarov* | *Back-end developer* |
| *Ivan Parashkevov* | *Scrum trainer* |
| *Georgi Kunev* | *Documentary* |
| *Stefan Ganchev* | *QA Engineer* |
| *Atanas Valchanov* | *Front-end developer* |

**1.Table of contents:**

1. **Contents**
2. **Topic**

**2.1*. Short Description***

**2.2*. Purpose***

**2.3*. Goals***

1. **Diagram**
2. **Progression of development**

**4.1. *Main steps***

**4.2*. Difficulties***

1. **Description of used functions**

**5.1. *Data Layer***

**5.2*. Presentation Layer***

**2.Topic:**

**2.1. Short Description:**

***Our program is a game called******Bletchley code breakers. It is a popular board game available also online provided by different suppliers. During the World War II a group of mathematicians was placed in Bletchley, UK. Their goal was to crack the German coding machine Enigma. We decided to show our take on the popular game and we hope you enjoy it.***

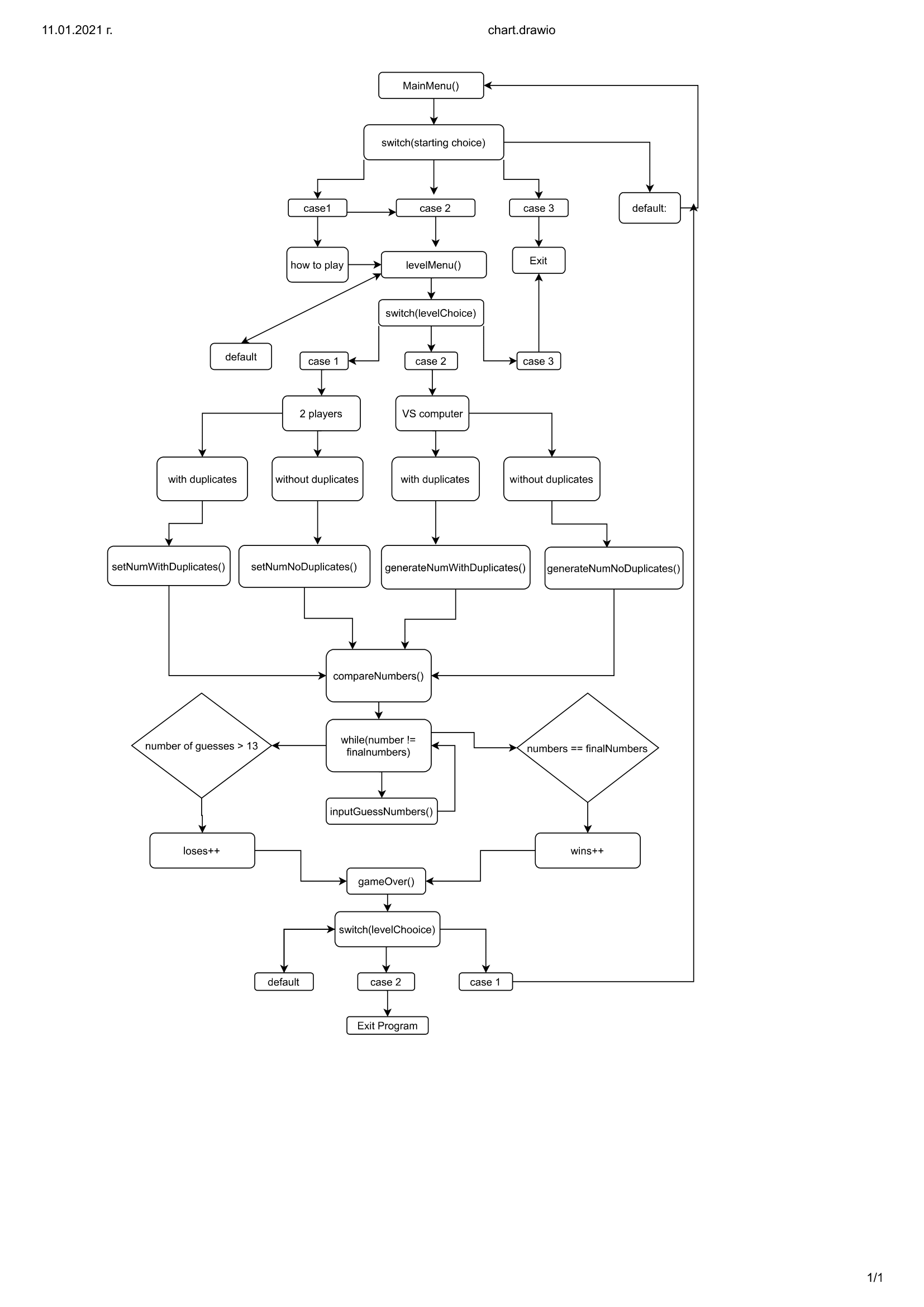
**2.2. Purpose:**

***The purpose of the game is to crack a 4-digit code within 13 tries or less. The player can choose to play either against the computer or against another player. If the computer Is chosen, he generates the numbers. If another player is chosen then the program asks him to choose the 4-digit code.***

**2.3Goals**

***Our goals are to make a fun and enjoyable game that can be played by one or two people. Future foals include translating the menus in different languages, adding more levels and optimizing the program.***

**3. Diagram:**



**4. Progress of development**

**4.1. Main steps in the implementation of the project**

* **Defining the main idea**
* **Create a Github repository and organize the tasks**
* **Write the back-end(logical) part of the code**
* **Add front-end(design) to the code so the user has feedback**
* **test the program and find bugs and report them to the dev team**
* **Make the documentation and the presentation of the project.**

**4.2. Difficulties we went through**

***One of the biggest difficulties we went through was the front-end of the project.The problem was that the information had to be divided in three columns. The solution we found was to use vectors instead of arrays, which made the whole process easier. Other difficulties we faced were restarting the game, making sure there aren’t any incorrect inputs from the user and clearing the console when needed.***

**Technologies used*: Visual Stuido Code, GitHub***

**5. Description of used functions:**

**5.1.Data Layer**

|  |  |  |
| --- | --- | --- |
| **void generateNumNoDuplicates** | **The computer generates 4 unique numbers.** | **int numbers[] – the 4-digit code the user tries to guess** |
| **void generateNumWithDuplicates** | **The computer generates 4 numbers which can repeat themselves.** | **int numbers[] – the 4-digit code the user tries to guess** |
| **bool checkInteger** | **Checks if the function’s argument is an integer value or not.If yes, returns true, otherwise false** | **String str – the string we check in the funtion** |
| **Void createVector** | **Adds a particular number of elements from an array to a vector** | **vector<int> arrayOfGuesses[] – array of vectors**  **int userNumbers[] – array we insert in the array of vector**  **int numberOfGuesses – index of the array wehre we are going to insert the vector**  **int vectorSize – size of the created vector** |
| **Void setNumNoDuplicates** | **The user is asked to input 4 unique numbers. Then they are stored in an array** | **int numbers[] – the 4-digit code the user tries to guess** |
| **Void setNumWithDuplicates** | **The user is asked to input**  **4 numbers which can repeat themselves. Then they are stored in an array** | **int numbers[] – the 4-digit code the user tries to guess** |
| **Bool areEqual** | **Checks if two arrays are the same. If yes, returns true. If no, returns false** | **int numbers[] - the 4-digit code the user tries to guess**  **int finalNumbers[] - array we compare to numbers[] to check if the user has guesses right** |
| **Void compareNumbers** | **Calls the inputGuessNumbers function, then adds the elements into a vector using CreateVector, and finally compared the inputed elements with the 4-digit code to check for similarities.** | **int numbers[] – the 4-digit code the user tries to guess**  **int userNumbers[] - array where we story the numbers from user input**  **int numberOfGuesses - number of guesses the user has made so far**  **int& wins – times the program has ended with a win**  **int& loses – times the program has ended with a lose** |

**5.2.Presenation Layer**

|  |  |  |
| --- | --- | --- |
| **Funtion Name** | **Description** | **Description of arguments** |
| **Void inputGuessNumbers** | **Gets 4 numbers as an input from the user** | **int userNumbers[] – array where we story the numbers from user input** |
| **Void outputVector** | **Outputs 3 vectors full with numbers** | **vector<int> guessedNumAndPos[] – vector with guessed numbers and positinos**  **int numberOfGuesses - number of guesses the user has made so far**  **vector<int> arrayOfGuesses[] – vector with all the guesses so far**  **vector<int> guessedNumbers[] – vector with numbers guessed but with wrong position** |
| **Int GameOver** | **Asks the user if he wants to continue the game. If yes, the function calls the starter function MainMenu. If no, the program is exited** | **No arguments** |
| **Void levelMenu** | **Outputs the level menu of the game and asks the user to input his level choice** | **int numbers[] – the 4-digit code the user tries to guess**  **int userNumbers[] - array where we story the numbers from user input**  **int numberOfGuesses - number of guesses the user has made so far**  **int& wins – times the program has ended with a win**  **int& loses – times the program has ended with a lose** |
| **Void mainMenu** | **Outputs the main menu of the game and asks the user to input his menu choice** | **int numbers[] – the 4-digit code the user tries to guess**  **int userNumbers[] - array where we story the numbers from user input**  **int numberOfGuesses - number of guesses the user has made so far**  **int& wins – times the program has ended with a win**  **int& loses – times the program has ended with a lose** |