



TEAM GRAVITY



Project for Scale Focus Sprint

Bletchley Game



Gravity



CONTENTS

1. Topic	2
2. Authors.....	2
2.1 Scrum Trainer – Radina Velichkova	2
2.2 Developer Front End – Martin Tsifondarov	2
2.3 Developer Back End – Kristina Bolashikova	2
2.4 Quality Engineer – Stanislava Andonova	2
2.5 Documentary – Dani Dimov	2
2.6 Code Checker – Aleksander Patrashkov	3
3. Our goals	3
4. Main stages of development and our contribution to the project	3
5.1 First stage.....	3
5.2 Second stage.....	3
5.3 Third stage	3
5. Level of difficulty	4
6. Used technologies	4
7. Description of the functions.....	6
Figure 1 - flowchart	6

1. TOPIC

The topic of our project was to develop a game known as Bletchley. The main goal of the game is to crack the Germans' battle ship coordinates by guessing them.

2. AUTHORS

2.1 SCRUM TRAINER – RADINA VELICHKOVA

- **RVVelichkova18@codingburgas.bg**
- Radina, our scrum trainer, took care of the distributing the roles of our team, organizing our meetings and assigning them to github.

2.2 DEVELOPER FRONT END – MARTIN TSIFONDAROV

- **MDTsifondarov18@codingburgas.bg**
- Martin, as our front-end developer, had to create the visual part of our program understandable in order to make it more accessible to the user.


2.3 DEVELOPER BACK END – KRISTINA BOLASHIKOVA

- **kvbolashikova18@codingburgas.bg**
- Kristina, in her role of our back-end developer, developed the logical part of our program by creating various functionalities.

2.4 QUALITY ENGINEER – STANISLAVA ANDONOVA

- **srandonova18@codingburgas.bg**
- The quality engineer role was given to Stanislava. Her task was to check the quality of our program. She notes her findings by using excel tables and work plan developed by her.

2.5 DOCUMENTARY – DANI DIMOV

- **dsdimov18@codingburgas.bg**
 - Dani made the documentation.
- 

2.6 CODE CHECKER – ALEKSANDER PATRASHKOV

- **AKPatrashkov18@codingburgas.bg**
- Aleks our code checker had to check the code for syntactic and logical errors and also had to post comments to help understanding the functionality of our program.

3. OUR GOALS

- Our first goal was to create a game that consist of several playing options - Player VS Player repeating digits, Player VS Player non-repeating digits, Player VS Computer repeating digits, Player VS Computer non-repeating digits, similar to Bletchley.
- We have multiple future goals, one of them is to add a new functionality that shows statistics of the games the user has played. We intend to make the interface more appealing and friendly for the user.

4. MAIN STAGES OF DEVELOPMENT AND OUR CONTRIBUTION TO THE PROJECT

5.1 FIRST STAGE - PLANNING

- Firstly, we formed our team, distributed our roles and registered our team under the name of Gravity. After separating the roles our scrum trainer assigned our task and we were ready to start working.

5.2 SECOND STAGE - REALIZATION

- After our tasks were assigned, we had short meetings if everything is going smoothly. We decided to have two sessions, first for the completing the code and second for quality checking it. During that time, we were also making our documentation and presentation.

5.3 THIRD STAGE - PRESENTATION

- The final part of our project was to have meeting where we discussed our program's functionality and make everything clear for our presentation. After that we were ready present in front of the judges the final product.

5. LEVEL OF DIFFICULTY

- This was the first time when we had to have a Quality engineer in the team, so it was quite new learning to understand the concepts of this position and the way things work. At the beginning we could not figure out how to implement automated testing and started a manual one but as some time passed, we finally figured it out.
- It was quite a hurdle for the developers to visualize the entering numbers from the first player (Player VS Player) with "*". Another challenging thing was to create a random generator for four non-repeating digits.
- We had several issues during the separation of the files because we had to configure them for the unit testing process.
- It was rather hard to configure the code in a way in which when we enter specific character(s) or number(s) (which ended up being the number 10) the program stops the process of guessing
- Another thing that the developers struggled with was figuring out where in the code was the mistake which caused the program to not be able to return back to the main menu after the third option from it had been previously selected.

6. USED TECHNOLOGIES

- We used Microsoft PowerPoint to create a presentation, with which we could show you our work.
- The documentation was made using Microsoft Word.
- Our Quality Engineer has used Microsoft Excel to create tables for each test case.
- Photoshop was used to edit images that we have used in our project.
- Visual Studio is an integrated development environment which our developers used to create the Bletchley-like game.
- In Git Hub we organized our meetings, monitored the progress of our project and submitted our changes.
- We used Discord as means of communication.

- C++ is the programming language we used for the development of our project.
- With Slide Go we were able to make our documentation and presentation more pleasing to the eye.



7. DIAGRAM

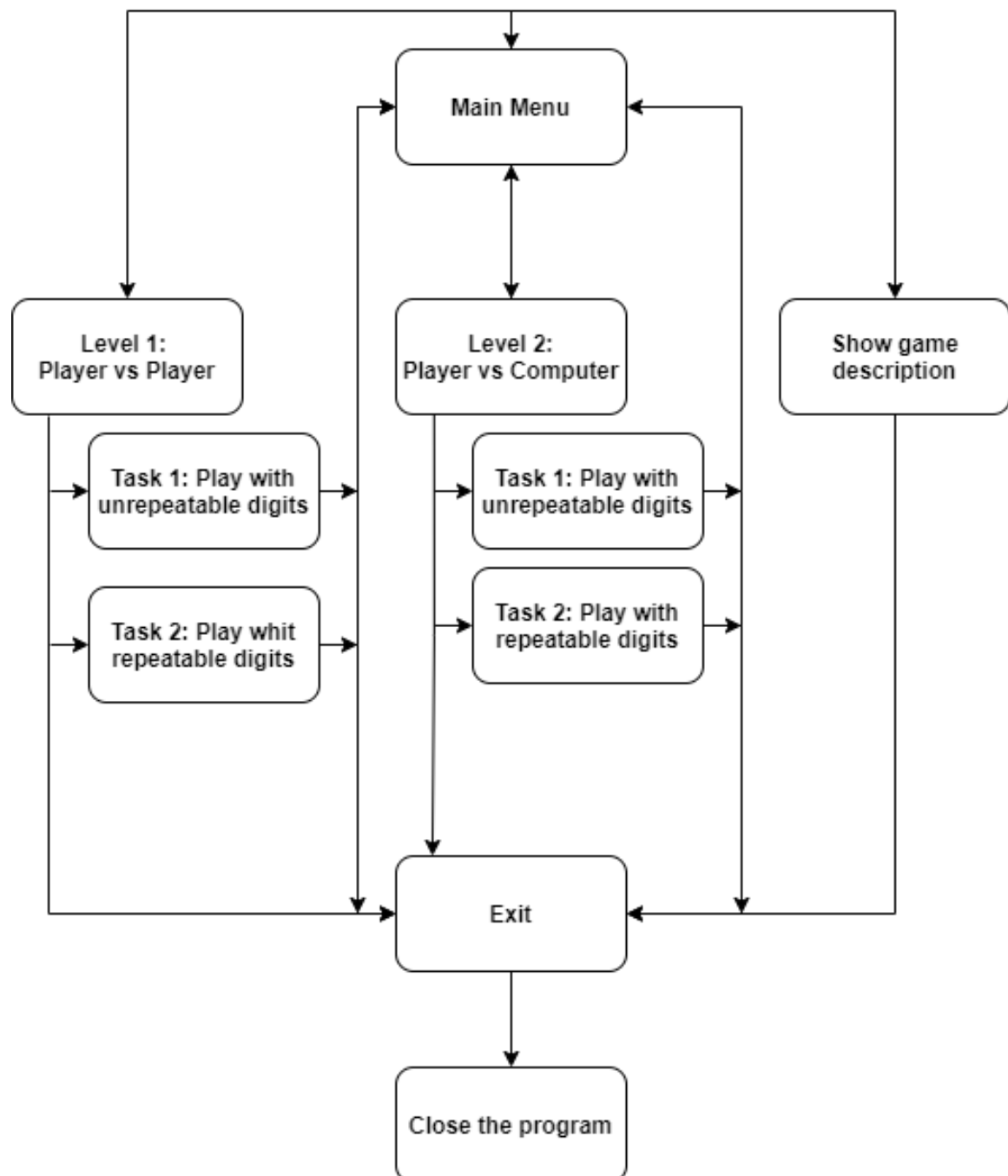


Figure 1 - flowchart

8. DESCRIPTION OF THE FUNCTIONS

Name	Type	Argument(s)	What it does
readInt	Int	None	Accepts one int value and outputs an error and restarts the input if it is of any other type
readIntGuess	Int*	The user's guess	Accepts four int values and outputs an error and restarts the input if it is of any other type
checkInRange	Bool	The user's guess	Checks if all the digits from the user's guess are in the predefined range (0 - 7)
checkForRepeatingNumbers	Bool	The user's guess / code to be guessed	Checks if there are any repeating numbers in the user's guess or in the entered code for guessing if the option for unrepeatable numbers has been selected
randomInt	Int	None	Generates a random int number between 0 and 7
randomNumberWithRepetition	Int*	The code about to be randomly generated	Generates a four-number code which can have repeating numbers
randomNumberNoRepetition	Int*	The code about to be randomly generated	Generates a four-number code which can't have repeating numbers

checkForSameNumberAndPosition	Int	The code, the user's guess	Counts how many numbers (in the same positions as in the code) the user has guessed
checkOnlyForSameNumbers	Int	The code, the user's guess	Counts how many numbers (NOT in the same positions as in the code) the user has guessed
enterNumbersWithRep	Bool	User's guess	Inputs the user's guess without checking for repeating numbers
enterNumbersNoRep	Bool	User's guess	Inputs the user's guess and checks for repeating numbers
enterHidden	Void	The code to be entered	Inputs the user's code without displaying it on the screen
enterHiddenNoRep	Void	The code to be entered	Inputs the user's code without displaying it on the screen and checks for repeating numbers
processGuesses	Void	The code, and a variable that indicates if the numbers can repeat or not	Counts the number of guessed numbers and numbers and positions for all the attempts and prints the result at the end
chooseOption	Int	None	Prints two options: to return to the main menu or to exit the program, and accepts an input
processlevelOne	Int	The code, and a variable that indicates if the numbers can repeat or not	Inputs the user's code and then starts to play level one of the game (player vs. player)

processlevelTwo	Int	The code, and a variable that indicates if the numbers can repeat or not	Generates a random code and then starts to play level two of the game (player vs. computer)
displayMainMenu	Void	None	Displays the main menu of the program and accepts an option
displayLevelOneOptions	Void	None	Displays the menu of the first level of the program and accepts an option
displayLevelTwoOptions	Void	None	Displays the menu of the second level of the program and accepts an option
displayGameOverview	Void	None	Displays the rules of the game and some tips
spaces	Void	An int number	Outputs spaces
displayGreeting	Void	None	Displays a greeting to the user at the initial start of the program