Logo

Description automatically generated

Table of Contents

[Table of Contents 2](#_Toc130741711)

[1.About Us 3](#_Toc130741712)

[2.Game Description 3](#_Toc130741713)

[3.Stages of Realisation 3](#_Toc130741714)

[4.Used Technologies 3](#_Toc130741715)

[5.Functions 4](#_Toc130741716)

# 1.About Us

|  |  |  |
| --- | --- | --- |
| Name | Role | Grade |
| Miroslav Ganev | Scrum trainer | 9A |
| Filipa Popova | Front-end developer | 9B |
| Alex Kazakov | Back-end developer | 9V |
| Polina Dineva | QA Engineer | 9G |

# 2.Game Description

The game consists of 3 levels, each with a different difficulty. When the game starts, the player enters their own name and selects the difficulty level. In the process of playing, the player solves tasks and answers questions related to bitwise operations. If the player indicates a correct answer, a life is taken from the enemy, but if he marks a wrong answer, a life is taken from his. The game is won if the player manages to answer enough questions correctly to take down the enemy.

# 3.Stages of Realisation

1st Week - Creating logo and discussing ideas about the game.

2nd Week - Making basic game’s design and creating our main character.

3rd Week - Implementing our game idea through code.

4th Week -Finishing the game and making documentation.

# 4.Used Technologies

* Visual Studio – for writing the code.
* Power Point – for creating the presentation.
* Word – for creating the documentation.
* Excel – for creating the QA documentation.
* Adobe – for creating the logo.
* GitHub – for the project synchronization.
* Git – to commit our changes.
* C++ - for creating the game.
* Raylib – for game visualization

# 5.Functions

|  |  |
| --- | --- |
| Function | Action |
| void drawCenterLines(); | draws lines that meet at the center of the screen |
| void selectRectangle(bool& flag, Rectangle rec, std::vector<bool\*>& v); | checks if the user has clicked on a given rectangle and changes the value of a given bool respectively |
| texture textureMake(Image\* img, int width, int height); | loads an image, resizes it and returns the image converted into a texture |
| void gravity(pos& playerPos, int pHeight); | acts as gravity for the player |
| void moveRight(pos& playerPos, int pWidth) | allows the player to move right |
| void moveLeft(pos& playerPos, int pWidth) | allows the player to move left |
| void showHealth(); | displays the players health |
| void movement(); | allows the player to move according to the user’s input |
| int getFloorHeight(); | returns the value of the variable floorHeight |
| texture\* getEnemy(); | returns a pointer to the enemy texture |
| pos\* getEnemyPos(); | returns a pointer to the enemy position |
| std::vector<question>\* getQuestions(); | returns a pointer to the questions vector |
| int getEnemyHealth(); | returns the enemy’s health |
| void showEnemyHealth(); | displays the enemy’s health |
| texture getBackgroundTexture(); | returns the background texture |
| void finishGame(); | shows the finished game screen |
| void attack(); | let’s the player attack the enemy by pressing KEY\_E and answering questions related to bitwise operations |