**TEAM NAME: BELL**

**Project name: Falling Blocks**

**Team members:**

* **Mihail Stefanov**
  + Contribution: Design of user interface and code structure, preparation of game menus, high score retrieval and update, mouse events and buttons, participation in team meetings, code refactoring and debugging;
* **Detelin Maslev** 
  + Contribution: Design of overall gameplay algorithm, including creation of Figure prototype, figure generation, movement, rotation, drawing, interaction with positioned figures, removal of full lines, next figure display, game over conditions, participation in team meetings, code refactoring and debugging;
* **Vladislav Petrov**
  + Contribution: Creation of Timer prototype, creation of keypress events and their functionality, including sideways figure movements and accelerated down movement, participation in team meetings and code debugging;
* **Edward Slavov**
  + Contribution: Implementation of scorekeeping functionality and score display, creation of a pause button, adding its functionality, including a pause message display, adding initial colors and borders to the figures and code debugging.

**Project purpose:** We have created a game similar to the popular game Tetris. The goal of the game is to guide the falling blocks, aiming to form a horizontal line of ten chained squares. Each line awards points, gets removed and the blocks above it move one square down. The game ends when the squares of the matrix approach its top. The player can navigate through the user interface via mouse clicks. The game is played by pressing the up, down, left and right buttons. At the end of the game, the player can submit the score achieved and compare the performance with other players on a high scores board. The game was written in JavaScript with minimal use of HTML and CSS (for the page positioning, canvas and score submission box) and AJAX (for score retrieval). The following diagram summarizes the game logic:

**Screen redrawn, if needed, every 20 milliseconds**

**Screen redrawn and updated, if needed, every 20 milliseconds**

**Control via key events**

**On game over condition**

**Calling initialize()**

**Mouse event**

**Calling showGameOptions()**

**Calling showGameHelp()**

**Calling beginGame()**

**Calling showHighScores()**

**Loading initial resources**

- Variables

- Functions

- Highscore

through

AJAX

Game Start Screen