Mazaya Rahman & Mahwish Jawed
CS428 - Final Project Milestone
Write Up
4/8/21

NOTE: Please open index.html to play our game

Task	Description
Generic cube constructor, test and **render** cubes, add cube colors	BoxGeometry, EdgeGeometry and MeshBasicMaterial are used to create cubes. Random x and z coordinates in a range are set for the cube. They are then added to the scene.
Set up website and game menu	Created a game menu page with a start button to start the game using **HTML** and **CSS**. Added an event listener to the main game code to start the game once the button on the menu page is clicked.
Initialize game objects and add to scene	Utilized **Three.js** to create the scene, camera, and renderer. Initialized an array of 100 cubes. Created an arrow geometry and applied a fill and edges. Also created a plane. The Cubes, arrow and plane are added to the scene. Then the animation function is called.
Animation	Each of the cubes z position is updated so they appear to move closer. Keyboard events for left and right move the cubes to left or right, and also tilts the plane and arrow accordingly. Checks for collision between each cube and arrow, and once they collide, it returns to the main menu. Once a collision is detected, game animation ends and the user is returned to the main menu screen. When the game start button is clicked, cubes are regenerated randomly again.
Collision detection	A check is done on whether the arrow's coordinates overlap with any of the cube's coordinates.
Keyboard Events	Added listeners for keyboard events for left and right keys.

Code Flow

1. User clicks **'Start Game'** button on the main menu
2. Event Listener detects the startGameButton was clicked and calls **initialize()** in **game.js**
3. **initialize()** adds arrows, plane, and cubes to the scene (these were initialized and declared as global variables, but the cube positions are generated within **initialize()**)
4. **initialize()** calls **animate()** once all objects are added to the scene
5. **animate()** hides the main menu if the game is in progress, otherwise displays it. **animate()** calls r**equestAnimationFrame()** and moves the camera, arrow, and cubes. **collisionCheck()** is called whenever a cube is moved closer to the user.
6. **collisionCheck()** returns true if the arrow and a cube have collided, else false.
    a. if returned true to **animate():** all objects from the scene are removed and **cancelAnimationReq()** is called. Since the game is over, the main menu pops up again.

Future Work

1. Will work more on game aesthetics: adding spotlight, shadows for cubes etc.
2. Keep track of current score and display it in game as it updates. Update and display the high score on the menu.
3. Add some levels of difficulty: increase speed as the game progresses etc.