I designed a space-themed game where the objective was to click on as many asteroids as possible without getting hit. There is a time limit of 20 seconds where the player can click as many asteroids as possible to score points. The game ends when 20 seconds are up or when the alien gets hit. So it is a game of concentration where a person will have to make sure they survive while trying to score as many points as they can.

I chose this because I could exercise most of the skills learned in this class such as making use of raphael elements and attributes, styling, sonifying my game, using while loops and arrays, initializing rates and positions, making use of random number generators, using getElementById to grab DOM objects and making use of event listeners for choosing the difficulty level.

I started with the template from our "click game" in class and one of my assignments. Using the layout page used in class, I styled the page with my own design to make it space themed.

First, I created the targets - the asteroids. The asteroids are basically 25 dots in an array. I made a while loop to ensure that each dot would have the same attributes and starting position so that it would seem like the asteroids are bursting out from the centre. However, I randomized the rate so that each asteroid would move at a different direction and speed.

The difficulty level is determined by how fast the dots are emitted. By using an event listener on the button, the speed of calling the emit function is changed.

I then created the alien and made it "draggable" to showcase one of the skills we learned in class. By tracking the mouse state, I made sure the alien can only be dragged if the mouse is on it.

The asteroids move with the moveTarget function as it updates the position and rates of the asteroids.

I calculated the distance between the asteroids and mouse clicks so that when an asteroid is clicked, it will disappear. The counter counts the clicked asteroid towards the score. dotArray[i].emittable = false ensures that once a dot is clicked, that particular dot will no longer be emitted again.

I also calculated the distance between the alien and the asteroid so that if the asteroid has not been clicked before and if it "hits" the alien, the game is over. If that asteroid has been clicked before, it will not affect the alien. Thus, the player has to be aware of the asteroid.

There is also the emitter function which is a new interval timer that emits by setting one dot's position back to the center of the page. If a "nextToEmit" dot has already been clicked (which can be tracked using dotArray[nextToEmit].emittable === false), the next "nextToEmit" dot will be called instead (by using the mod operator). If that dot has never been clicked, it gets called back to the centre of the screen. Furthermore, once an asteroid has been clicked, it cannot "destroy" the alien.

Initially, I wanted to randomize the position of the "nextToEmit" dots if they have never been clicked. But after testing it out, I think it makes the game too challenging to be fun. Therefore, I decided to call the dots back to the centre of the screen.

The game only starts when the player clicks start. When the game starts, the start button is hidden and the targets are shown. The timer also starts and the targets start moving when the start button is clicked. The ready function ensures that the game can go back to its original state if the game were to be restarted. There is also a game stop function that ensures everything is stopped when the game ends and that the game will restart so that the player can play again.