## Assignment Requirements

### Assignment 1: Image Carousel

**Description:** An image slider library.

**Requirements:**

1. Should have HTML content in the following format.

| <div class="carousel-container">  <img src="#" alt="Image" />  <img src="#" alt="Image" />  <img src="#" alt="Image" />  ... </div>  *<!-- OR -->*  <div class="carousel-container">  <div class="carousel-image-wrapper">  <img src="#" alt="Image" />  <img src="#" alt="Image" />  <img src="#" alt="Image" />  ...  </div> </div> |
| --- |

1. Should have functional controls like **forward** and **back** arrows and **indicator dots**.
2. Should have sliding animation when transitioning from one image to another.
3. When reaching a boundary, the slider should either change direction or animate towards the other boundary and continue in the same direction.
4. Indicator dots should be clickable and transition to the image indicated when clicked.

### Assignment 2: Configure timings in carousel

1. Transition and hold times should be configurable
2. Should automatically slide from one image to another with a fixed interval.
3. The page should allow multiple instances of the slider.
4. Should be responsive.
5. OOP using ES5

### Assignment 3: Box Collision and Ant Smasher

**Description:** Balls or boxes moving in random directions with collision detection between walls and each other.

**Requirements:**

1. Should be a rectangular boundary box
2. Should have instances of balls generated within the box. OOP using ES5
3. Balls should have rectangular collision detection with boundary walls of the box and should bounce back.
4. Balls should have rectangular/circular collision with other balls and should bounce back maintaining correct directions.
5. Balls should not overlap when colliding.
6. Balls should not go outside of the boundary.
7. Balls should have variable direction. When initialized, all balls should start moving in different directions.

**Extra Features:**

1. Variable ball count, speed, and sizes.
2. Implementation of elastic collision or some sort of physics-based collision that takes into account speed, mass, and angle of balls.
3. Stress Test: Test with 1000 balls. Expect minimal Lag.

**Ant Smasher Requirements:**

1. Boxes/Balls should be replaced by images of ants.
2. Ants should be destroyed when clicked by the mouse.

**Ant Smasher Extra Features:**

1. Ants should be animated gifs so that they look like they are moving.
2. Ants should be facing in the direction they are moving towards.

### Assignment 4: Car Lane Game

**Description:** A top-down 2D game with a three-lane road. The player car can switch lanes. The objective is to not collide with other cars on the road.

**Requirements:**

1. Should have 3 lanes.
2. Should have assets of car and road.
3. At any moment, other vehicles should be moving downward. Should look like the player car is moving faster.
4. The player car should be static in the y-direction
5. Using arrow keys or A/D keys, players can change the x-direction of the car, effectively changing lanes of the car.
6. The player cannot move outside the bounds of the road.
7. When changing lanes, the car should be positioned in pre-configured positions on the road (on the center of the lane and not in between lanes.)
8. When moving past an obstacle (a car), a score value should increase.
9. If the player car collides with another car, the game is over and the score should be reset.
10. Should have an initial start screen with a start button that initializes the game.
11. All three lanes of the road should not be blocked by generated obstacles. There should always be one lane empty for the player to pass the obstacle.
12. The speed of the game should increase over time.
13. Obstacle cars should always be generated so as to not give an unfair disadvantage to the player.
    1. The gap between obstacles should be big enough for the player to move past.

**Extra Features:**

1. Cars should animate toward a destination lane
2. The high score should be maintained and should persist.

### 

#### Car Game - Bullet implementation

**Requirements:**

1. The player car can have a mounted weapon that can throw projectiles straight ahead.
2. Ammo management should exist. (Limited Ammo, powerups on the road that increases ammo).
3. If a shot projectile hits an obstacle, the obstacle should be destroyed leaving open space for the player to stay in the lane.

**Extra features:**

1. Multiple instances of Game should be playable simultaneously.
2. Two player game with two cars.
   1. One car should not collide with another when changing lanes.
   2. Both cars should have all the features implemented for a single player game.

### Assignment 5: Flappy Bird

**Description:** Clone of the Flappy Bird game

**Requirements:**

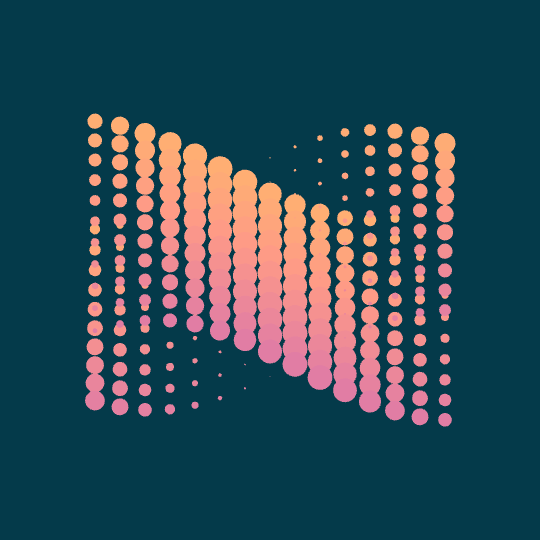
1. Should meet the bare requirements of Flappy Bird.
2. Should have assets of bird, pipe, and background.
3. Should maintain and persist scores and highscore.
4. Should have an initial start screen with a start button that initializes the game.
5. During a game over situation, the game should restart on a click of a Restart button without the need to reload the page.
6. The pipe generation should be fair to the player.
7. Should have animated bird assets.
8. The bird should accelerate downwards to the ground-based on gravity, instead of the y coordinate changing by a fixed value.

**Extra Features:**

1. Bird position change should animate when jumping, e.g. instead of decreasing the y value by a jump amount, gradually animate the jump.
2. Multiple instances of the game should be playable simultaneously.
3. The bird should gradually rotate so as to be facing the ground while falling downward.

### Assignment 6: Helix Animation

**Description:** Recreate an animation in Javascript



To be done in **Canvas**.