Simple Scrolling Game

For this project, you'll demonstrate an understanding of drawing and animation using the HTML5 canvas element as well as an understanding of JavaScript basics. You'll make a simple scrolling game using the HTML **canvas** element and the **requestAnimationFrame** method of the window object.

In the game, your player character is on the left edge of the screen and moves parallel to that edge when the player presses the relevant arrow keys. (Eventually we'll update the game to control the player character by tilting the mobile device.) Two types of objects will scroll across the screen toward the player. If the player comes into contact with a **harm** object, the number of lives remaining is reduced. If the player comes into contact with a **benefit** object, the points are increased.

You should use JS ES6 (e.g., function syntax with fat arrows; creating variables with let and const as appropriate, etc.)

This should be installable as a PWA.

We'll start parts of this in class, but you'll need to finish it on your own.

A few requirements, beyond those basics:

- Score, lives and level should be displayed on the canvas.
- The game should have a start screen explaining how to play.
  (It could be a separate div or it could just be some text written to the canvas.)
- When lives are reduced to zero, a "Game Over" message should be displayed and the animation should stop (not just be hidden.)

- Harm and benefit objects should respond to collision with player and with left edge by returning from the right side of the canvas, in a random y coordinate.
- Your game should have a theme, implemented as a background image in the game and as images for the player and harm/benefit objects.

  (**Note** that images are drawn on the canvas using the x,y as the top-left position, while the arc paths use the x,y coordinates as the center. This means you'll need to make some adjustments to your collision calculations; you can still use the distance formula, but you'll have to calculate an offset from the top-left of your images to determine the center.)
- The game should get harder across levels by increasing the "speed" with which harms move across the screen and by adding an additional harm object. Let score thresholds determine level change (e.g., 0 points = level 1; 100 points = level 2; etc)
- Data for each object (player, harm, benefit, game?) should be stored as JS Objects.

The following are not required, but for those who have the time and interest in making their game more interesting (in case it's something you want to show off), here are a few ideas.

1. Add a scrolling background
2. Add multiple scrolling backgrounds for a parallax background.
3. Animate one or more of your game objects.
4. Let the player pause the game by pressing a specific key.
5. Add a "timer"; give the game object a "frames" property that you increment inside the animation loop.
6. Have multiple harm and benefit objects; create an array of them. You could add more harm objects as the level increases or at timer thresholds. (This could be simplified by having one gameCharacter object type, which has as one of its properties, it's "role": harm, benefit, player, ??)

7. Add sound effects.
8. Let the player earn "protection" or "powers" at certain point values or by grabbing specific benefit items; implement this by changing the player's image or display, then check the state of the player's display upon collision. Let the protection or powers expire after a certain period of time (using the "frames" mentioned above) or after a certain number of collisions.
9. Track the player's "health" and display on the canvas.
10. Let the player fire projectiles at the objects. This is the same functionality as the harm and benefit objects, but the projectile moves *from* the player and when it hits the opposite edge, you don't need to return it to the other side.

Deliver the requirements before you start working on optional items, and don't get bogged down with optional items - post questions on Piazza as needed.

Project Delivery:

- Page should be viewable as a GitHub pages site, at **/it202-spr22-project2/**

Some possibly useful resources:

- https://gamedevelopment.tutsplus.com/tutorials/parallax-scrolling-a-simple-effective-way-to-add-depth-to-a-2d-game--cms-21510
- https://luka712.github.io/2018/12/02/Parallax-scrolling/
- https://www.codeandweb.com/texturepacker/tutorials/how-to-create-a-sprite-sheet