Game Design:
What story do I want tell?
What do I want the player to feel?

Algorithms Overview:
Input data
Logic statements
Output data

JavaScript Overview:
variables
statements
functions
if statements
lists and loops
classes and objects

Display:
Cartesian plane
HTML canvas template
Drawing on the canvas
Animating on the canvas
Layers

Game Loop:
update state
draw/render state

Game Objects:
Position
Velocity
Properties/states
Collisions

Game World:
Art (Tilemaps)
Encoding world

Code Organization:
Encapsulation
Abstraction

Game Art:

Animations with spritesheets
Creating game art/animations with piskelapp
Loading art

Audio:
Playing sounds and music
Loading audio

Advanced:
Saving/loading game state
Serverside vs clientside

Notes:

What to do when you need to know how something works in html/javascript?
Search https://developer.mozilla.org/en-US/ or w3schools.com

Canvas API and libraries: https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API

Game tutorials:

Example game:
https://developer.mozilla.org/en-US/docs/Games/Tutorials/2D_Breakout_game_pure_JavaScript
Game loop: https://developer.mozilla.org/en-US/docs/Games/Anatomy

Use www.jsfiddle.net for experimenting.

Examples:

Start drawing on HTML canvas:

```
<html>
  <body>
    <canvas id="canvas" width="480" height="320"></canvas>
  </body>
</html>
```

```
const canvas = document.getElementById('canvas');
const ctx = canvas.getContext('2d');

ctx.fillStyle = 'green';
ctx.fillRect(10, 10, 150, 100);
```

```
var canvas = document.getElementById("myCanvas");
var ctx = canvas.getContext("2d");

ctx.beginPath();
ctx.rect(20, 40, 50, 50);
ctx.fillStyle = "#FF0000";
ctx.fill();
ctx.closePath();

ctx.beginPath();
ctx.arc(240, 160, 20, 0, Math.PI*2, false);
ctx.fillStyle = "green";
ctx.fill();
ctx.closePath();

ctx.beginPath();
ctx.rect(160, 10, 100, 40);
ctx.strokeStyle = "rgba(0, 0, 255, 0.5)";
ctx.stroke();
ctx.closePath();
```

```
// the triangle
context.beginPath();
context.moveTo(100, 100);
context.lineTo(100, 300);
context.lineTo(300, 300);
context.closePath();

// the outline
context.lineWidth = 10;
context.strokeStyle = '#666666';
context.stroke();

// the fill color
context.fillStyle = "#FFCC00";
context.fill();
```

Add a game loop:

```
// From https://www.sitepoint.com/quick-tip-game-loop-in-javascript/
function update(progress) {
  // Update the state of the world for the elapsed time since last render
}

function draw() {
  // Draw the state of the world
}

function loop(timestamp) {
  var progress = timestamp - lastRender

  update(progress)
  draw()

  lastRender = timestamp
  window.requestAnimationFrame(loop)
}
var lastRender = 0
window.requestAnimationFrame(loop)
```

3d math stuff:

Projecting a 3d scene onto a 2d scene:
https://en.wikipedia.org/wiki/3D_projection
How is the view-matrix der

Multiplying matrices:
https://www.mathsisfun.com/algebra/matrix-multiplying.html

3d math primer:
https://www.3dgep.com/category/math/
https://www.3dgep.com/understanding-the-view-matrix/

Quaternions solve gimbal lock problem:
https://www.3dgep.com/understanding-quaternions/

Snake Game:
https://www.khanacademy.org/computer-science/new-program/6246451551813632

Castle game art:

https://3docean.net/item/modular-dungeon-set-allinone-package/233335
Animate outline first
Add color/shading afterwards