TASK 7-8 FINAL REPORT: JSON, Arrays & Visualization

This report presents the full implementation, improvements, and outputs for Task 7-8. It covers the use of JSON data, array methods in JavaScript, creative visual representation, interactive features, and console output results.

1. JSON Data Loading

The iris.json file is loaded using the Fetch API and parsed into an array of 150 objects. Each object includes: sepalLength, sepalWidth, petalLength, petalWidth, and species.

2. JavaScript Array Methods with Output

map() - Add a 'color' field randomly to each iris

```
const irisesWithColors = data.map(iris => ({
  ...iris,
  color: possibleColor[Math.floor(Math.random() * possibleColor.length)]
}));
```

-> Output: Each object has a new 'color' field from the palette.

filter() - Keep irises with sepalWidth < 4

```
const filteredIrises = irisesWithColors.filter(iris => iris.sepalWidth < 4);
console.log("Filtered Irises (sepalWidth < 4):", filteredIrises);
```

```
console.log("Filtered Irises (sepalWidth < 4):", filteredIrises);

[
  { sepalLength: 5.1, sepalWidth: 3.5, petalLength: 1.4, ... },
  { sepalLength: 4.9, sepalWidth: 3.0, petalLength: 1.4, ... },
  ...
  118 items total
]
```

reduce() - Calculate average petalLength

```
const totalPetalLength = irisesWithColors.reduce((sum, iris) => sum + iris.petalLength, 0);
const averagePetalLength = totalPetalLength / irisesWithColors.length;
console.log("Average Petal Length:", averagePetalLength.toFixed(2));
```

-> Output: 3.76

find() - First iris with petalWidth > 1.0

```
const findPetalWidth = irisesWithColors.find(iris => iris.petalWidth > 1.0);
console.log("Find petalWidth > 1.0:", findPetalWidth);
```

-> Output: {
  "sepalLength": 6.3,
  "sepalWidth": 3.3,
  "petalLength": 6.0,
  "petalWidth": 2.5,
  "species": "virginica",
  "color": "#d35d3f"
}

some() - Check if any iris has petalLength > 10

```
const somePetalLengthMoreThan = irisesWithColors.some(iris => iris.petalLength > 10);
console.log("Some petalLength > 10:", somePetalLengthMoreThan);
```

-> Output: false

some() - Check if any iris has petalLength == 4.2

```
const somePetalLengthEqual = irisesWithColors.some(iris => iris.petalLength == 4.2);
console.log("Some petalLength == 4.2:", somePetalLengthEqual);
```

-> Output: true

every() - Check if all irises have petalWidth < 3

```
const everyPetalWidthLessThan = irisesWithColors.every(iris => iris.petalWidth < 3);
console.log("Every petalWidth < 3:", everyPetalWidthLessThan);
```

-> Output: true

every() - Check if all irises have sepalWidth > 1.2

```
const everySepalWidthMoreThan = irisesWithColors.every(iris => iris.sepalWidth > 1.2);
console.log("Every sepalWidth > 1.2:", everySepalWidthMoreThan);
```

-> Output: true

toSorted() - Sort irises by petalWidth ascending

```
const irisesWithColorsSorted = irisesWithColors.toSorted((a, b) => a.petalWidth - b.petalWidth);
console.log("Sorted Irises:", irisesWithColorsSorted);
```

-> Output: Array sorted from petalWidth 0.1 to 2.5

3. Visualization & Interactivity

Each iris object is visualized as a floating, breathing coloured circle on a canvas. The size represents petalLength, and the color is randomly assigned from a palette.

User Interactions:

- Hover to reveal full object data (species, petal/sepal sizes)
- Accurate info display even with overlapping circles
- Buttons to filter: Setosa, Versicolor, Virginica, Petal Length > 5
- Start/Stop motion toggle with label update
- Clean UI styling for buttons, background, and layout

We visualized each iris as a floating and breathing coloured circle using canvas. Each circle's size reflects its petal length, and hovering allows the viewer to explore more about the different flower's details like species, petal and sepal dimensions. We wanted to symbolize a living organism through a humanistic approach, applying the real to the digital. As they grow and shrink they follow the rhythmic dance of what our lungs can make.

4. Improvements Based on Feedback

- Hovering logic fixed to avoid incorrect multiple data displays
- Individual buttons for each species
- Toggle motion button label now changes properly
- Font enlarged and info panel spacing improved
- Added visual polish to background and controls

5. Conclusion

The assignment has been completed successfully with thoughtful additions and refinements.