Advanced JavaScript Tasks

1. Deep Object Comparison  
Write a function `deepEqual(obj1, obj2)` that compares two objects for deep equality, checking nested objects, arrays, and primitives.

2. Promise Chain Error Handling  
Create a series of promises using `.then()` that will handle both success and failure conditions. Each promise should either resolve or reject randomly. Ensure that rejected promises are caught and handled properly.

3. Throttling Function  
Implement a `throttle(fn, limit)` function that ensures the function `fn` is only called once every `limit` milliseconds, even if it is triggered multiple times.

4. Custom Event Emitter  
Create a basic event emitter class from scratch. It should have methods like `on()`, `emit()`, and `off()`.

5. Virtual DOM Implementation  
Implement a simplified version of a virtual DOM diffing algorithm. It should be able to compare two tree-like structures (representing DOM elements) and return the changes needed to update the actual DOM.

6. Debounce Function  
Write a `debounce(fn, delay)` function that delays the execution of the function `fn` until after `delay` milliseconds have passed since the last time it was invoked.

7. Recursive Tree Traversal  
Implement a recursive function `traverseTree` to traverse a tree structure and log each node. The tree nodes have `name` and `children` properties.

8. Custom `bind()` Implementation  
Implement your own version of the `bind()` function that works the same as the built-in JavaScript method.

9. Implementing a Memory Game  
Create a memory card game where cards are randomly shuffled, and the user has to match pairs. If two cards are flipped and don't match, they flip back after a short delay.

10. Circular Linked List  
Implement a circular linked list with functions to add nodes, remove nodes, and traverse the list.