**Backtracking Algorithms in JavaScript:**

Backtracking is a technique used to solve problems by incrementally building a solution and backtracking when the current solution cannot be extended further. It explores all possible options and prunes the search space by discarding partial solutions that cannot lead to a valid solution. Backtracking algorithms are commonly used for problems that involve searching for a feasible solution in a large search space.

**Characteristics:**

* Incremental Construction: Backtracking algorithms construct a solution incrementally by making decisions at each step and backtracking when a dead end is reached.
* Exploration of Search Space: Backtracking explores the entire search space by recursively exploring all possible options and discarding invalid or unpromising solutions.
* Optimization: Backtracking algorithms often involve pruning techniques to reduce the search space and improve efficiency by avoiding unnecessary exploration.

**Example Problems:**

N-Queens Problem:

The N-Queens problem involves placing N queens on an NxN chessboard such that no two queens attack each other. Backtracking is commonly used to explore all possible configurations of queens on the board and backtrack when a conflict is detected.

Subset Sum Problem:

The Subset Sum problem involves finding whether there exists a subset of a given set with a specified sum. Backtracking techniques can be used to explore all possible subsets of the set and backtrack when the sum exceeds the specified target.

**Implementation:**

Backtracking algorithms are typically implemented using recursion, where each recursive call explores a potential solution and backtracks when the solution cannot be extended further. Pruning techniques such as constraint propagation and early termination are used to improve efficiency by avoiding unnecessary exploration of the search space.