**JavaScript String Reversal Functions**

# Function `reverse1(str)`: Using Built-in JavaScript Methods

Purpose:

* To reverse a string efficiently using JavaScript's native array methods.  
  Implementation Details:  
  - **`split('')**`: Converts the string into an array of characters.  
  - `**reverse()`:** Reverses the order of elements in the array.  
  - `**join('')`:** Merges the elements of the array back into a single string.  
  This function is ideal for simple use cases where a quick and easy solution is required for string reversal.
* Code:

function reverse1(str) {

    return str

      .split('')  // Convert string to array of characters

      .reverse()  // Reverse the array elements

      .join('');  // Join the array back into a string

}

* Example Usage:

let word = 'mara';

console.log(reverse1(word));  // Output: 'aram'

# Function `reverse2(str)`: Iterative Approach

Purpose:

* To demonstrate a manual iterative method for reversing a string, suitable for educational purposes or scenarios where built-in methods are to be avoided.  
  Implementation Details:  
  - Initialization: A new empty string `result` is prepared to store the reversed string.  
  - Loop: A `for` loop runs from the end of the string back to the start, appending each character to `result`.  
  This approach gives more control over the process and avoids the overhead of array operations.
* Code:

function reverse2(str) {

    let result = '';

    let length = str.length;

    for (let i = length -1; i >= 0; i--) {

        result +=str[i];

    }

    return result;

}

* Example Usage:

let word = 'makan malam';

console.log(reverse2(word));  // Output: 'malam nakam'

# Function `reverse3(str)`: Modern JavaScript Syntax

Purpose:

* To utilize modern JavaScript language features (like the `for...of` loop) for reversing a string, enhancing code readability and maintainability.  
  Implementation Details:  
  - Loop: The `for...of` loop iterates through each character of the string.  
  - String Prepend: Each character is prepended to the resultant string `reversed`, effectively reversing the order of characters.  
  This method is particularly useful for developers who prioritize clean and expressive syntax.
* Code:

function reverse3(str) {

    let reversed = '';

    for (let character of str) {

      reversed = character + reversed;

    }

    return reversed;

}

* Example Usage:

let word = 'dia';

console.log(reverse3(word));  // Output: 'aid'