```
/**
 * Function to calculate the Fibonacci sequence up to the nth number
 * @param {number} n - The number of Fibonacci terms to generate.
 * @returns {Array} - An array containing the Fibonacci sequence up to the nth term.
 *
 * @example
 * fibonacci(5);
 * // Returns: [0, 1, 1, 2, 3]
 *
 * @description
 * The Fibonacci sequence starts with 0 and 1, with each subsequent number being the sum of the two preceding ones.
 * This function generates the sequence iteratively and returns the result as an array.
 */
function fibonacci(n) {
    // Check for invalid input
    if (n <= 0) {
        throw new Error("Input must be a positive integer.");
    }

    let sequence = [0, 1]; // Initializing the sequence with the first two numbers.

    // Generate the Fibonacci sequence up to the nth term
    for (let i = 2; i < n; i++) {
        sequence.push(sequence[i - 1] + sequence[i - 2]);
```

```
    }

    return sequence.slice(0, n); // Return the sequence truncated to the nth term
}

/**
 * Function to check if a given number is prime.
 * @param {number} num - The number to check.
 * @returns {boolean} - Returns true if the number is prime, otherwise false.
 *
 * @example
 * isPrime(5);
 * // Returns: true
 *
 * @description
 * A prime number is a number greater than 1 that has no divisors other than 1 and itself.
 * This function checks if a given number is prime by testing divisibility from 2 to the square root of the number.
 */
function isPrime(num) {
    if (num <= 1) return false; // Numbers less than or equal to 1 are not prime

    for (let i = 2; i <= Math.sqrt(num); i++) {
        if (num % i === 0) {
            return false; // If divisible, it's not prime
```

```
        }
    }
    return true; // The number is prime if no divisors were found
}

// Example usage
try {
    console.log(fibonacci(10)); // Outputs the first 10 Fibonacci numbers
    console.log(isPrime(7));    // Outputs: true
} catch (error) {
    console.error(error.message); // Catches and logs any errors
}
```