# JavaScript Promises - Hands-On Practice Questions

These exercises are designed to strengthen your understanding of JavaScript Promises. They cover creating promises, chaining, error handling, and working with async/await. Focus on writing code to solve each problem.

* 1. Create a promise that resolves with the message 'Data Loaded Successfully' after 2 seconds. Log the resolved value.
* 2. Create a promise that randomly either resolves with 'Success' or rejects with 'Error Occurred'. Handle both outcomes using .then() and .catch().
* 3. Create a promise chain that:  
   - Resolves with the number 5  
   - Multiplies it by 2 in the next .then()  
   - Adds 10 in the next .then()  
   - Logs the final result.
* 4. Write a function fetchUserData() that returns a promise resolving to {id: 1, name: 'Alex'} after 1s. Chain another promise to fetchUserPosts() that returns ['Post1', 'Post2'] after 1s. Log user details along with posts.
* 5. Use Promise.all to run three promises that resolve after 1s, 2s, and 3s with values 'A', 'B', and 'C'. Log the combined result.
* 6. Use Promise.race to run two promises: one resolves after 2s with 'Fast', another after 4s with 'Slow'. Log the result.
* 7. Use Promise.any with three promises where the first two reject and the third resolves. Log the resolved value.
* 8. Convert the problem in Question 4 (fetchUserData and fetchUserPosts) into async/await syntax.
* 9. Write an async function that waits for a promise to resolve after 3s with 'Process Complete'. Log the result after awaiting.
* 10. Write an async function that uses try/catch to handle a rejected promise (reject with 'Something Went Wrong').
* 11. Simulate an API call getWeather(city) that returns a promise resolving with `Weather in <city>` after 2s. Call it with 'London' and log the result.
* 12. Create a function getStockPrice(stock) that returns a promise resolving with `<stock>: 100` after 1s. Use Promise.all to get prices for ['AAPL', 'GOOG', 'TSLA'] and log them together.
* 13. Simulate fetching data from two APIs using async/await. The first resolves after 2s with 'Users Data', the second after 3s with 'Orders Data'. Log both results once they are available.