Fibonacci

A fascinating number sequence that finds applications in biology and art.

Problem Statement

There are exactly two rules for the fibonacci sequence:

1. The first two numbers are 1.

2. Aside from that, every subsequent number in the sequence is the sum of the previous 2.

For example, the first few numbers of the sequence are 1, 1, 2, 3, 5, 8, 13.

Write a function named fib that takes a number, n, and returns the nth number in the sequence. For example, if I called fib with the number 6 it should return the number 8.

Suitability

Students that deal with this problem in class may have an advantage over those who didn't. Still, the sequence is simple enough to explain compactly. Experience with the sequence shouldn't be too substantial an advantage.

Implementation Quirks

Most students are taught to solve this problem recursively and then iteratively as a way of understanding both recursion and performance concerns.

The simplest recursive solution is tiny. The simplest iterative solution is hardly longer, but more complicated to reason about.

Interview Waypoints

State the problem clearly and consistently every time using the **Problem Statement** above.

- The output can't be cheated. If applicant strays to finding novel relationships between the input and output give them a minute then redirect them to the problem statement.
- Once they believe they have a working solution have them work through a couple of inputs. Be sure to use small input numbers, especially for testing recursive solutions.
- If they produce the iterative solution then provide the recursive solution for discussion.
- If they produce the recursive solution discuss it's performance, complexity, and maintainability. Then ask them to produce the iterative solution.

Expanding Beyond JS

Once the function has been defined you can quickly probe into their comfort with assembling web pages by asking them to expand their solution into a simple web page.

Problem Statement:

Given the function you wrote please put it in it’s own file. Write the markup for an html page that when viewed in the browser simply says, “The sixth fibonacci number is 8.” The page should load your reusable function and call it with the number 6 and put the result into the DOM in the proper place. Finally, link a CSS file to the page. In the CSS file write a rule that causes the “8” in the rendered sentence to be red.

Implementation Quirks

Expanding the problem into a simple web page gives you the chance to see how comfortable the applicant is making a complete web page. Do they understand the mechanics? Having the syntax memorized isn’t necessary, but do they know the kinds of tags and attributes required?

You might ask them how they would go about finding the proper syntax. Where would they get their boilerplate from?

Conclusion

The Fibonacci sequence isn’t a terribly challenging problem. Still, it involves enough fundamental concepts to be a fair judge of how fluent they are in JavaScript. It is also a good jumping off point for discussing execution semantics, performance, recursion, looping constructs, zero versus one based sequences, off-by-one errors, html boilerplate, favorite language references, and framework dependence.

Ideas for Improvement

- Update the question to be in the context of a web component.
- Tweak the sequence rules to try and level the playing field between those who have and haven’t done the sequence before.