OUTPUT BASED

INTERVIEW QUESTION-141

JS

Follow on

@Duvvuru Kishore

Interviwer: Can you explain why the following code **outputs 3, 3, 3** when the loop condition is **i < 3**?

```
for (var i = 0; i < 3; i++) {
  setTimeout(function() {
    console.log(i);
  }, 1000);
}
```

Explanation 👉

The Role of var

var is function-scoped, all iterations of the loop share the same i variable.

Therefore, when the setTimeout callbacks execute, they all reference the same i, which is 3 by the end of the loop.

Loop Mechanics

Initialization:

The loop starts with `i = 0`.

Condition Check and Iteration:

The loop condition `i < 3` is checked. If true, the loop body executes.

Increment Step:

After the loop body executes, the increment expression `i++` is evaluated.

Iteration Details

First Iteration:
i = 0: The condition 0 < 3 is true, so the loop body executes.
After the body executes, i is incremented to 1.

Second Iteration:
i = 1: The condition 1 < 3 is true, so the loop body executes.
After the body executes, i is incremented to 2.

Third Iteration:

i = 2: The condition 2 < 3 is true, so the loop body executes.

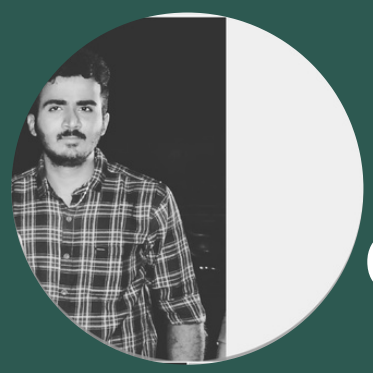
After the body executes, i is incremented to 3.

Exit Condition:

Now, i = 3. The condition 3 < 3 is false, so the loop exits.

Final State

By the time the loop exits, i has been incremented to 3 in the last step of the loop. This is why, when the setTimeout callbacks execute, they see the final incremented value of i, which is 3.



Follow on

@Duvvuru Kishore