The ternary operator

An overview of the ternary operator which also serves as a control structure

Notice you already know another way to return values without `if-else`:

```
function getLampColor( state ) {
    return state == 1 ? 'Red' : 'Green';
}

console.log(getLampColor(1));
console.log(getLampColor(5));
```

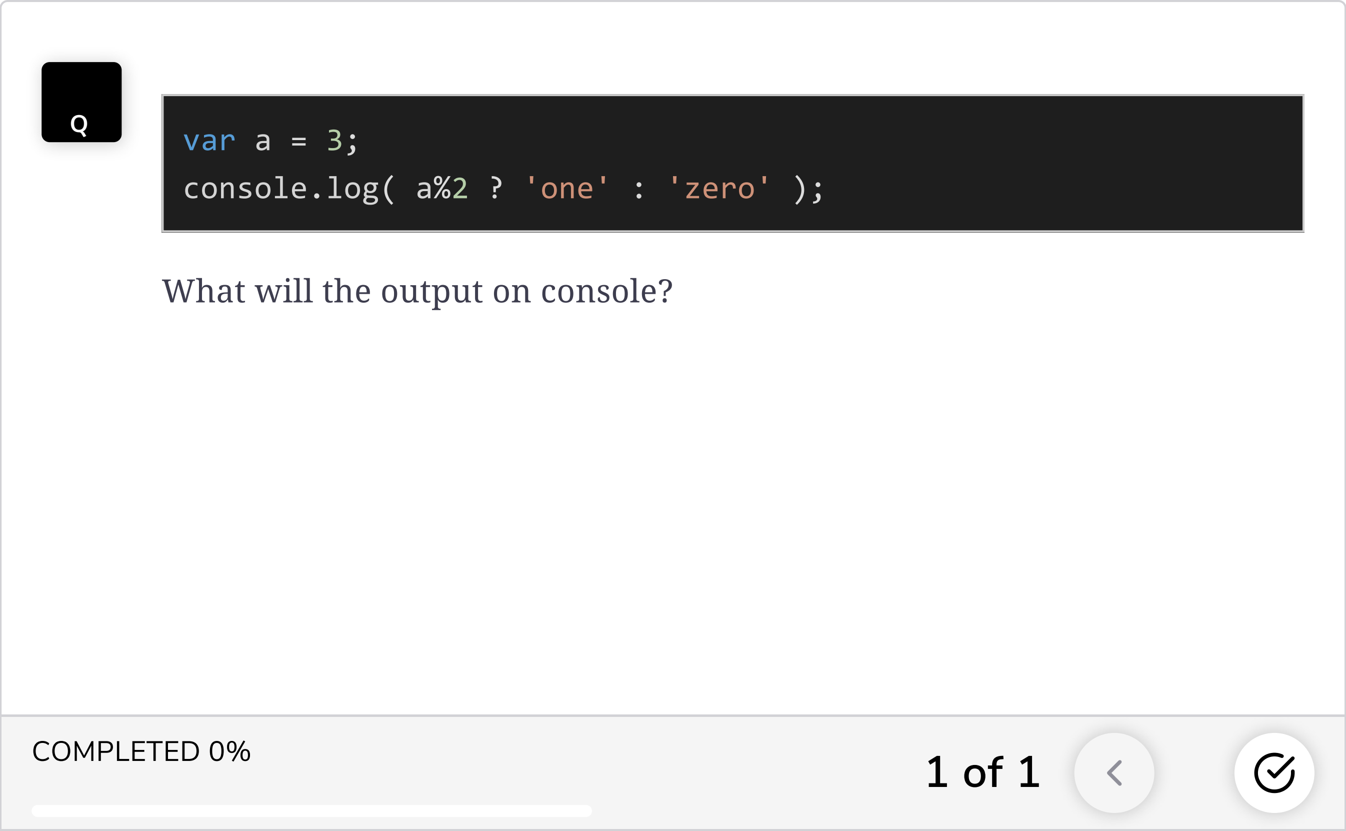
The ternary operator gives you red if the state is `1`, and green otherwise.

We can place another ternary expression in place of `‘Green’` to add more lamp states:

```
function getLampColor( state ) {
    return state == 1 ? 'Red' :
           state == 2 ? 'Yellow' :
           state == 3 ? 'Green' :
           'Error';
}

console.log(getLampColor(1));
console.log(getLampColor(2));
console.log(getLampColor(3));
console.log(getLampColor(4));
```

Some people don’t like this way of writing code. You can hit up their rant about how awful this construct is under the name of *ternary operator abuse*. I personally do not share their opinion, but this does not matter. If you believe you should use it, use it.



```
var a = 3;
console.log( a%2 ? 'one' : 'zero' );
```

What will the output on console?