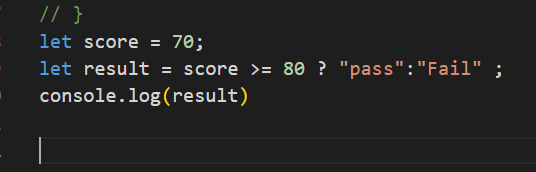
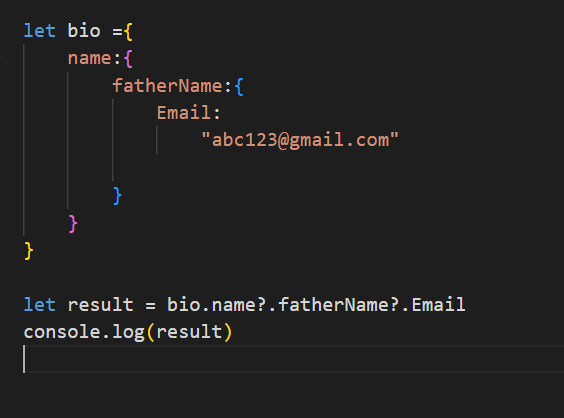
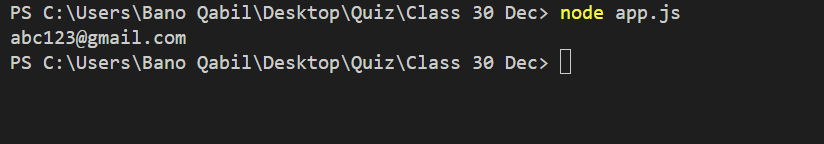
**1. Rewrite the following code using a ternary operator:  
let result;  
if (score >= 80) {  
    result = "Pass";  
} else {  
    result = "Fail";  
}**  


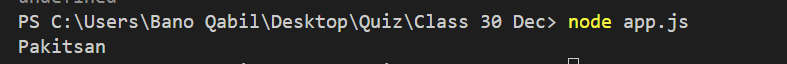


**2. How does the optional chaining operator (?.) work, and how can it be used to access nested properties of an object?**

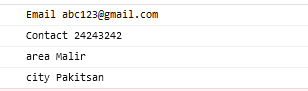


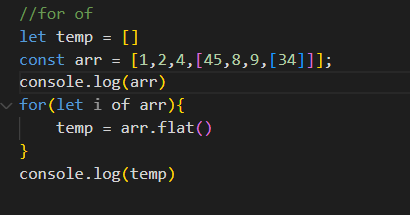


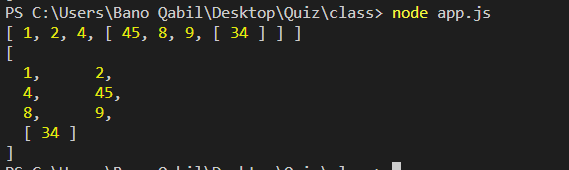


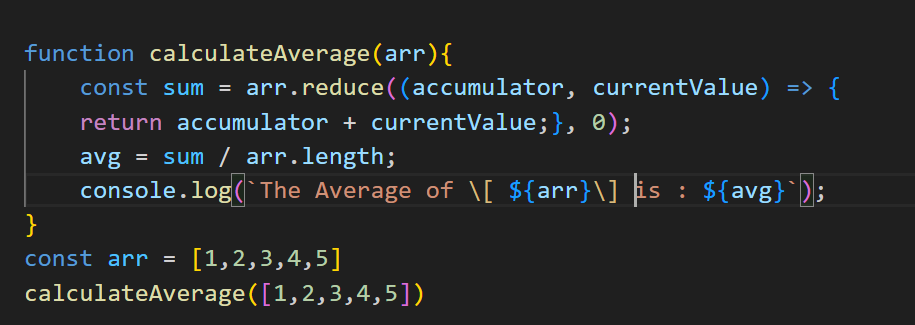
  
  
3. **Compare the for...in loop and the for...of loop in terms of their use cases and the types of values they iterate over.**





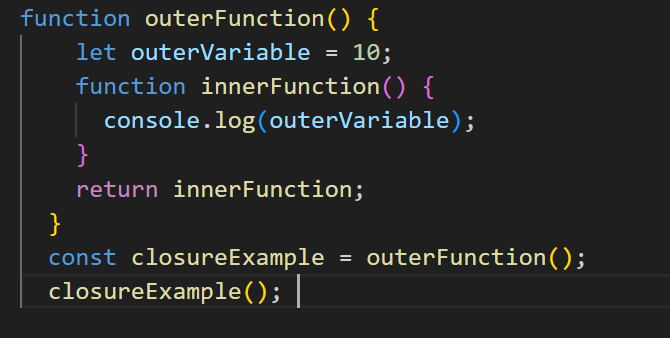


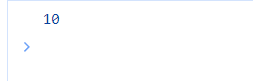
  
 **4. Define a function calculateAverage that takes an array of numbers as an argument and returns the average value.**

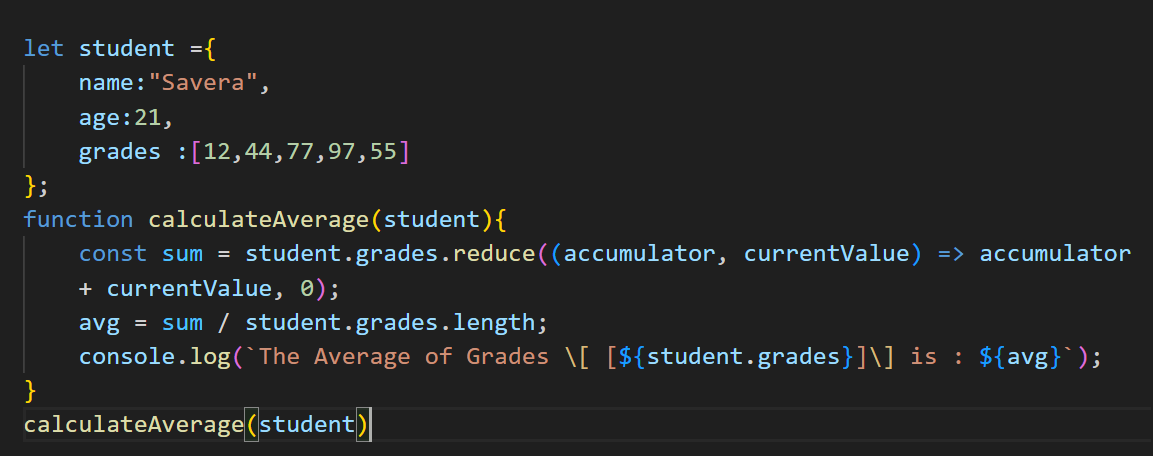




**5. Explain the concept of "closures" in JavaScript and provide an example of their practical use.**



  
 **6. Create an object named student with properties name, age, and grades. Add a method calculateAverage that calculates the average of the grades.**

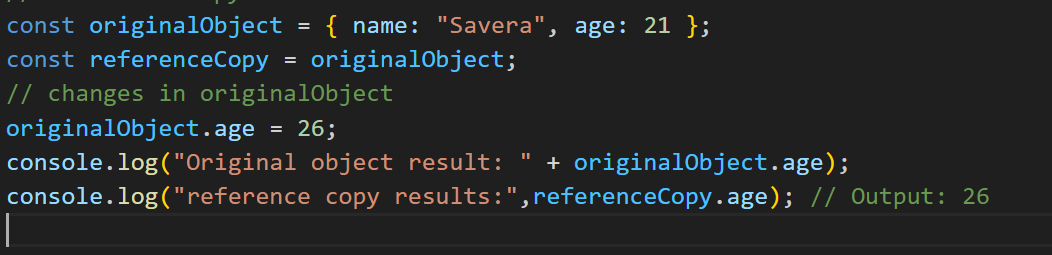


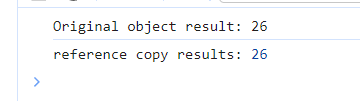


**7. How can you clone an object in JavaScript and also give one example each deep copy, shallow copy, and reference copy**

**Reference Copy.**

In a reference copy, the new object is just another reference to the same object in memory. Changes made to one object will affect the other.

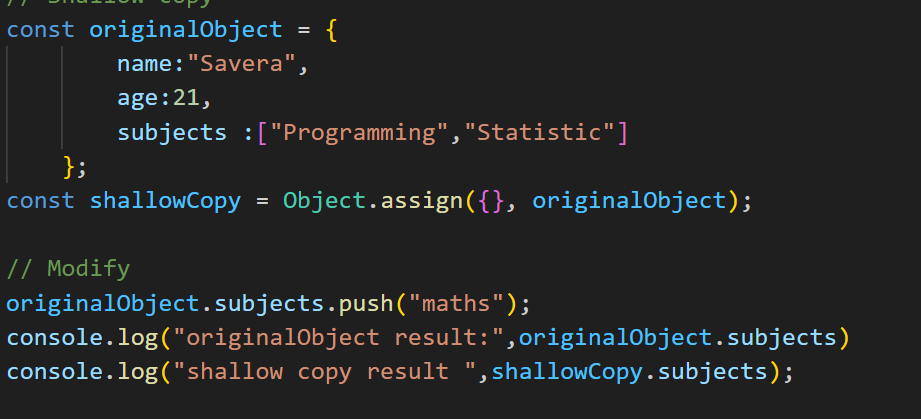


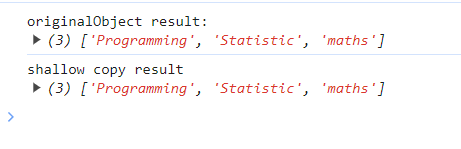


**Shallow Copy**

In a shallow copy, a new object is created, and the properties are copied. However, if the object contains nested objects, only references to those nested objects are copied, not the nested objects themselves.

If we change one the another is also change because both objects references the same array in memory.

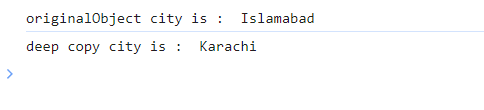




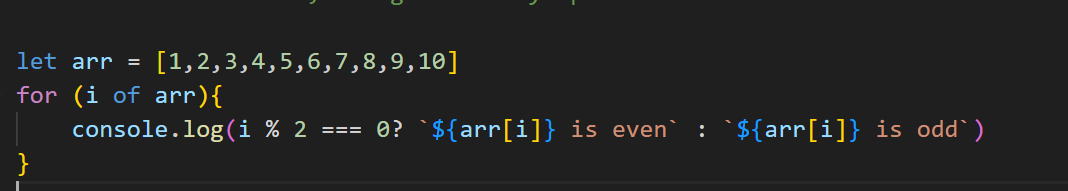
**Deep Copy**

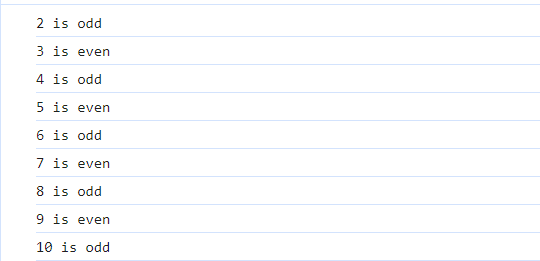
A simple way to perform a deep copy in JavaScript is to use the combination of JSON.stringify and JSON.parse





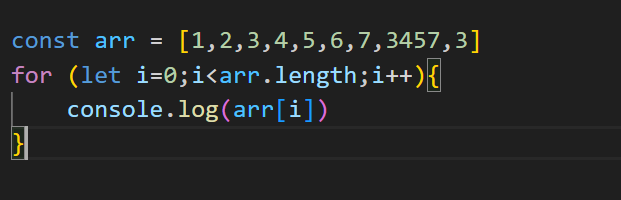
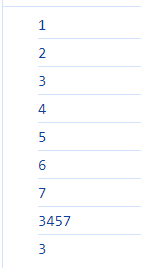
**8. Write a loop that iterates over an array of numbers and logs whether each number is even or odd, using a ternary operator.**

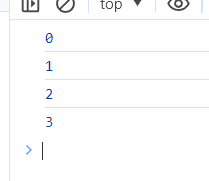
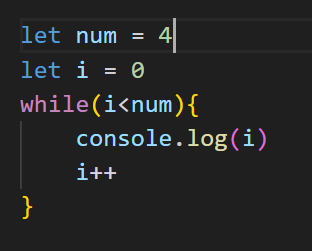


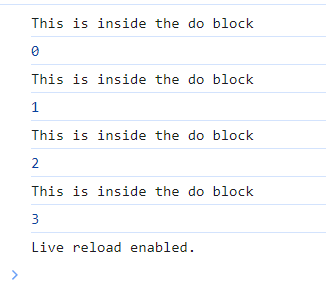
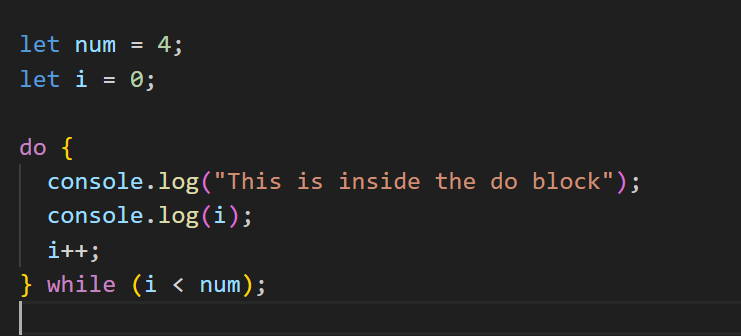


**9. Describe the differences between the for loop, while loop, and do...while loop in JavaScript. When might you use each?**

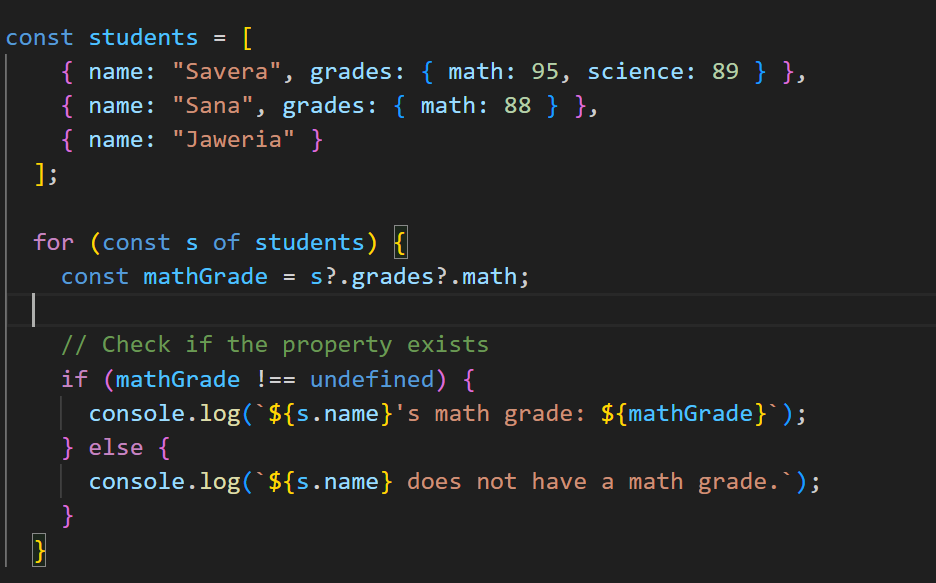
* **Use for loop**:
  + When the number of iterations is known.
  + When iterating over arrays or a specific range.
* **Use while loop**:
  + When the number of iterations is not known beforehand.
  + When the loop should continue as long as a certain condition is true.
* **Use do...while loop**:
  + When you want to ensure the loop body is executed at least once.
  + Similar to **while** but with a guaranteed initial execution.

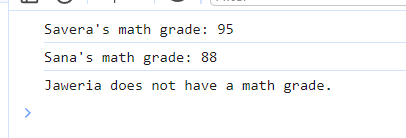


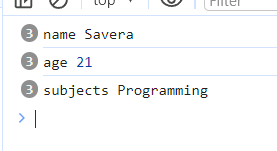
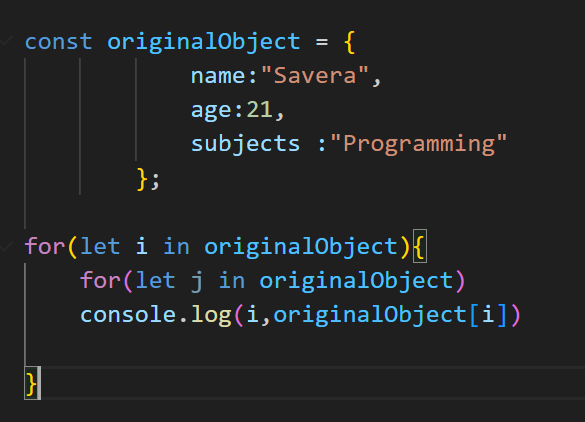




**10. Provide an example of using optional chaining within a loop to access a potentially missing property of an object.**



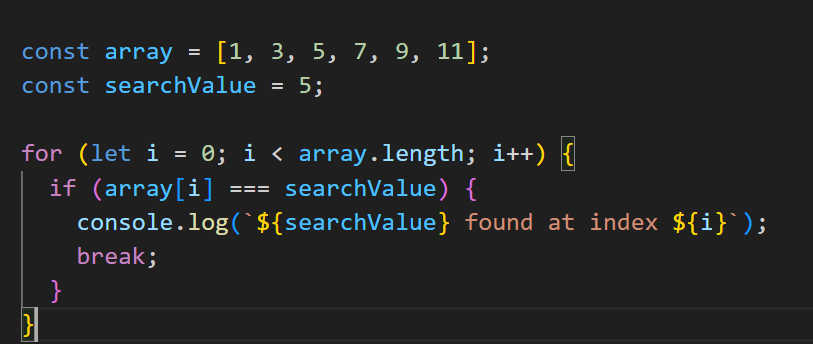


**11. Write a for...in loop that iterates over the properties of an object and logs each property name and value.**  


**12. Explain the use of the break and continue statements within loops. Provide scenarios where each might be used.**

**Break:**

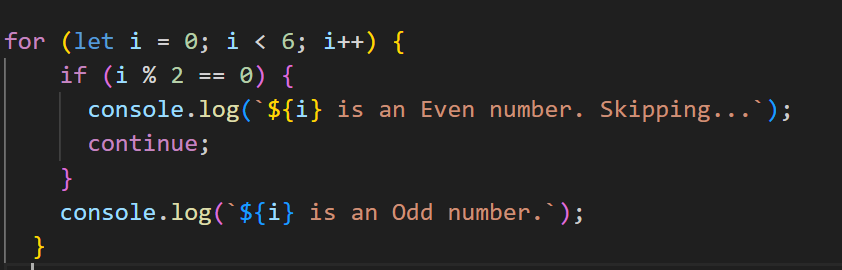
it immediately exits the loop, and the program continues with the next statement after the loop. Breaking out of a loop when a specific condition is met. Avoiding unnecessary iterations when a certain condition occurs.

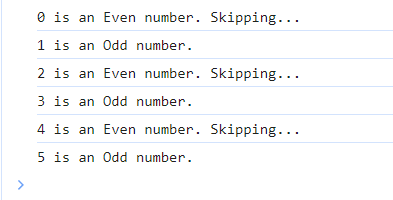




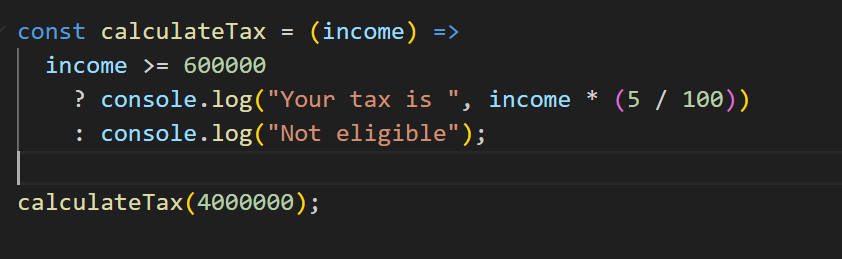
**Continue:**

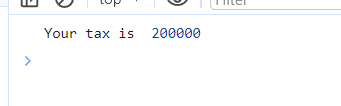
The continue statement is used to skip the rest of the code inside a loop for the current iteration and move on to the next iteration

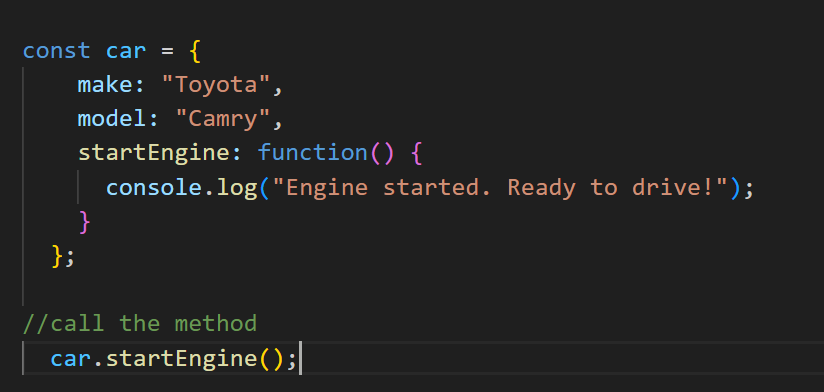


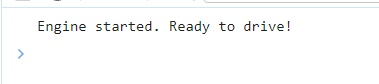


**13. Write a function calculateTax that calculates and returns the tax amount based on a given income. Use a ternary operator to determine the tax rate.**



 **14. Create an object car with properties make, model, and a method startEngine that logs a message. Instantiate the object and call the method**.



  
  
**15. Explain the differences between regular functions and arrow functions in terms of scope, this binding, and their use as methods.**

