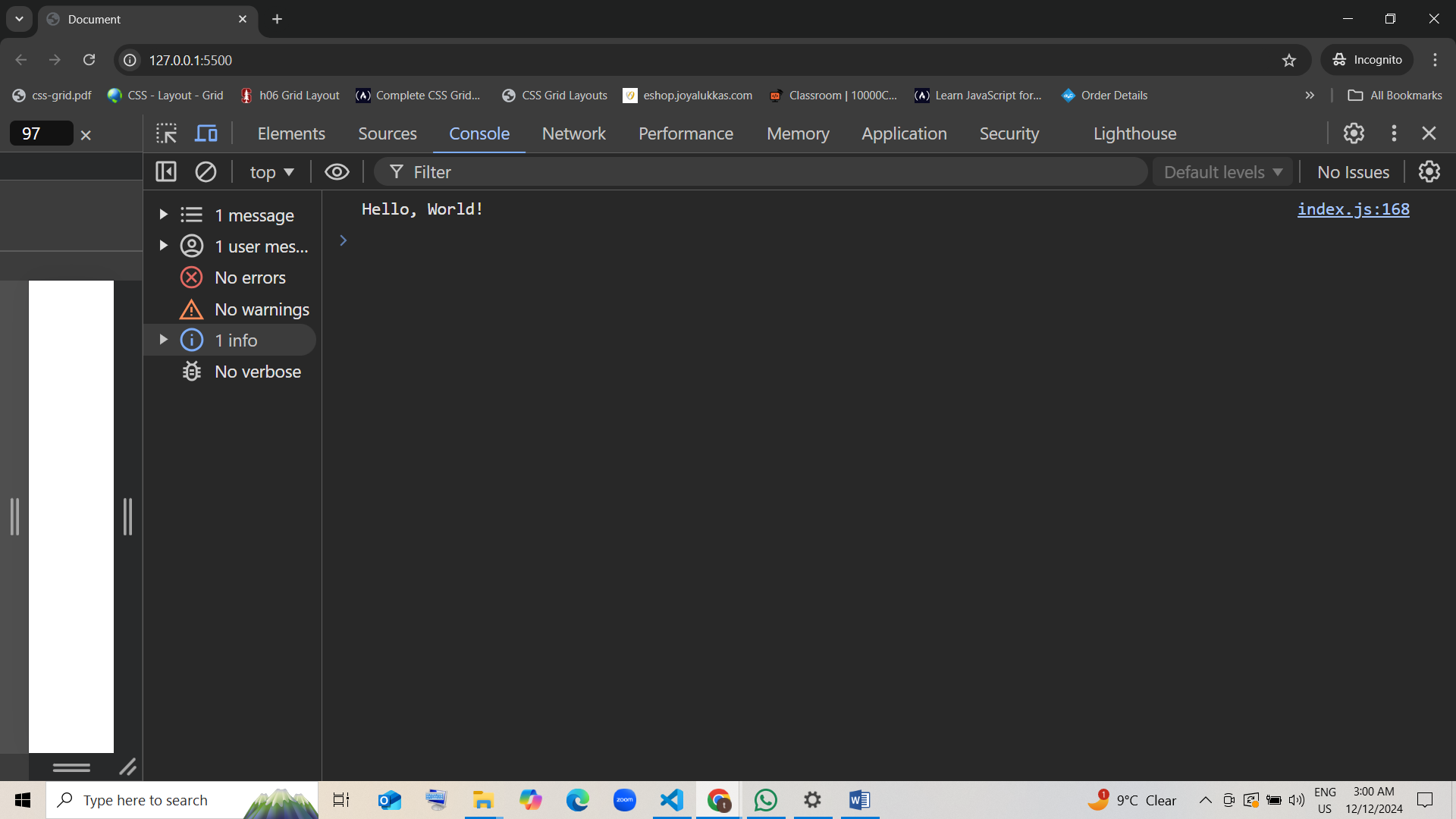
# JavaScript Function MCQs

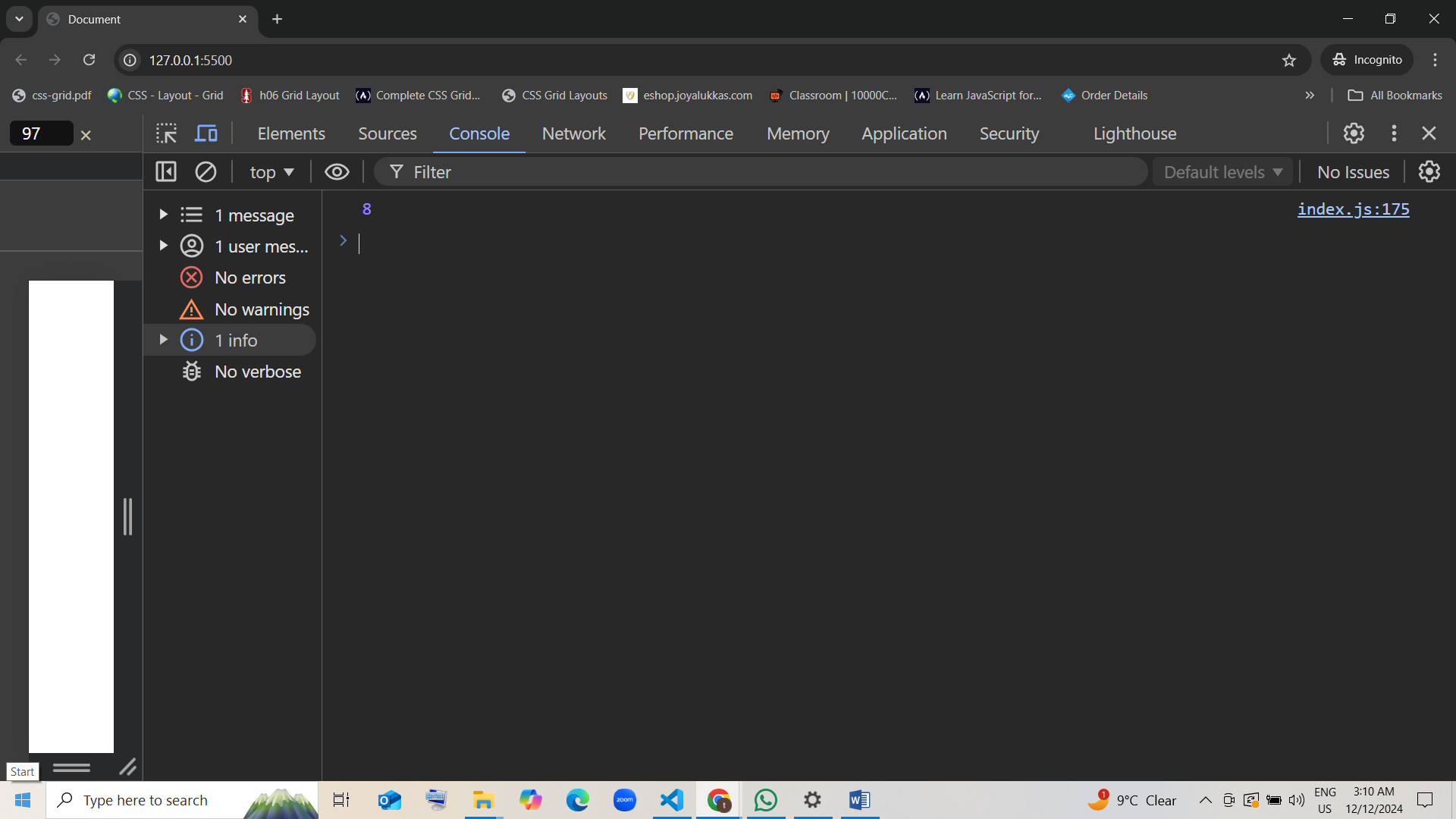
1. **Function Call with Simple Return**

* function getMessage() {  
   return "Hello, World!";  
  }  
    
  console.log(getMessage());
* What will be the output of the code above?
  1. Hello, World! Guess [correct]
  2. undefined
  3. Hello
  4. Error



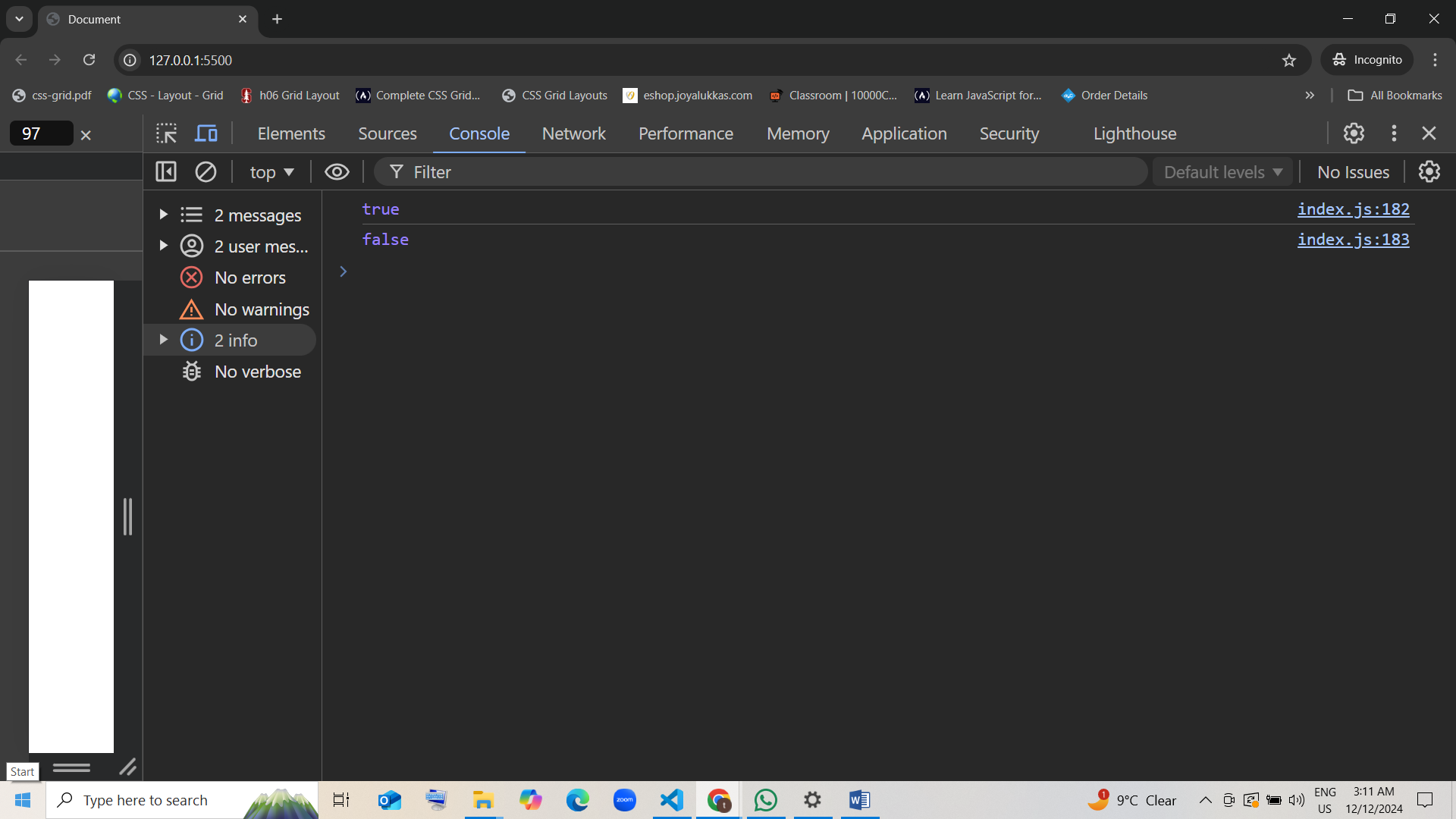
1. **Function Call with Parameters**

* function addNumbers(a, b) {  
   return a + b;  
  }  
    
  console.log(addNumbers(3, 5));
* What will be the output of the code above?
  1. 8 guess [correct]
  2. 35
  3. 53
  4. Error



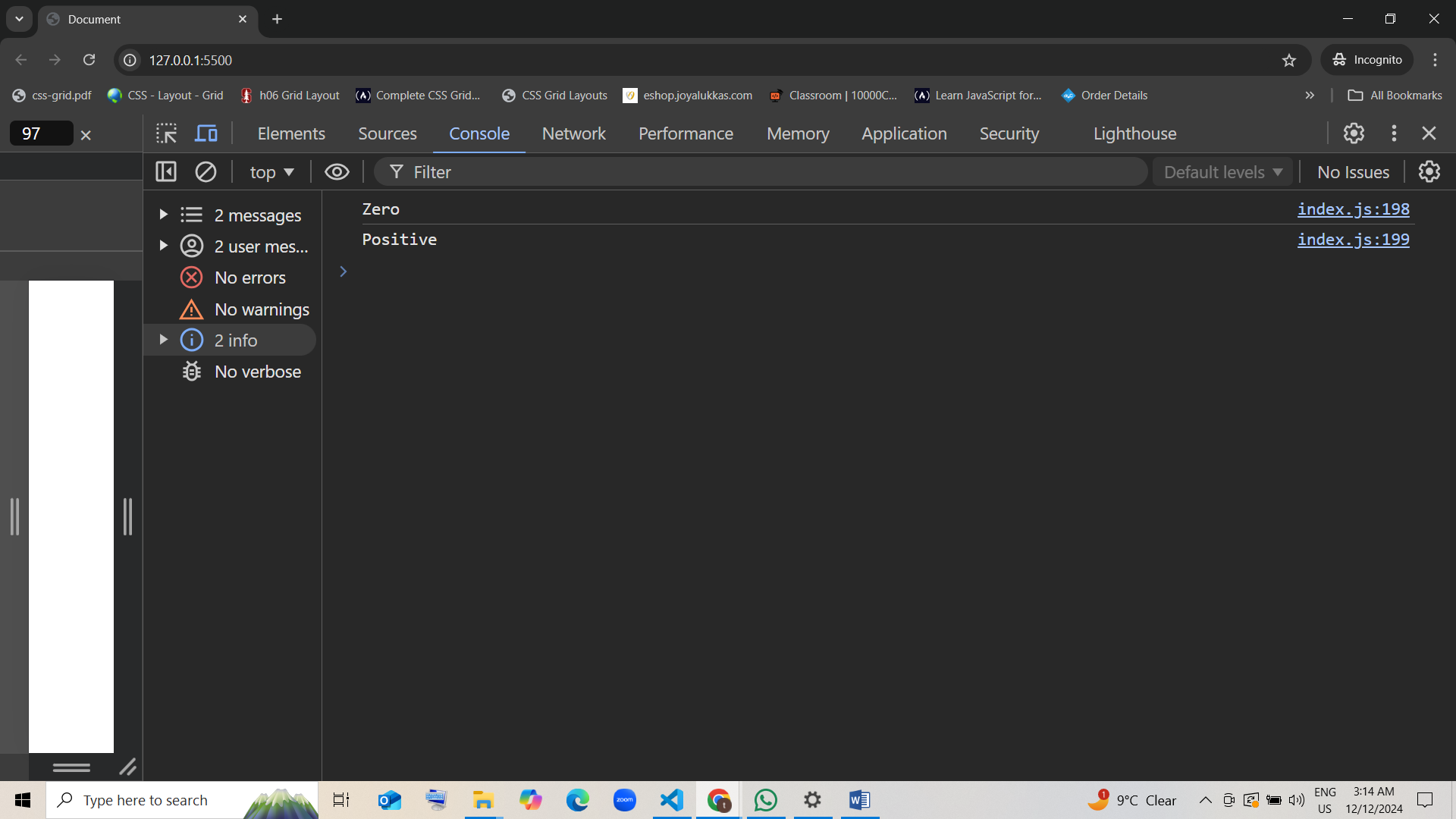
1. **Function Return with Boolean Check**

* function isEven(num) {  
   return num % 2 === 0;  
  }  
    
  console.log(isEven(10));  
  console.log(isEven(7));
* What will be the output of the code above?
  1. true and false guess [correct]
  2. false and true
  3. true and true
  4. false and false



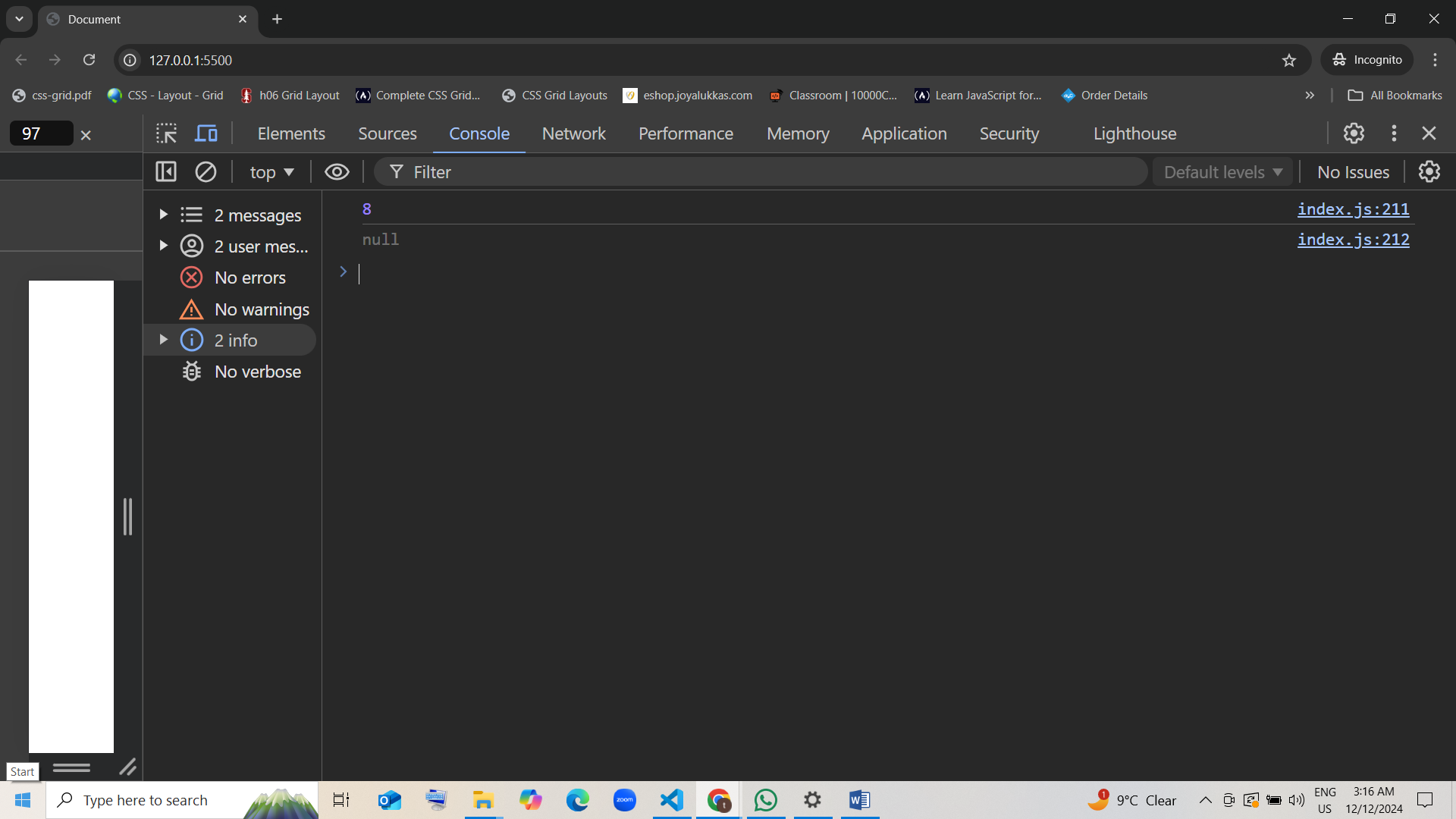
1. **Function with Multiple Return Statements**

* function checkSign(num) {  
   if (num > 0) {  
   return "Positive";  
   } else if (num < 0) {  
   return "Negative";  
   } else {  
   return "Zero";  
   }  
  }  
    
  console.log(checkSign(0));  
  console.log(checkSign(5));
* What will be the output of the code above?
  1. Zero and Positive guess [correct]
  2. Zero and Negative
  3. Positive and Zero
  4. Negative and Positive



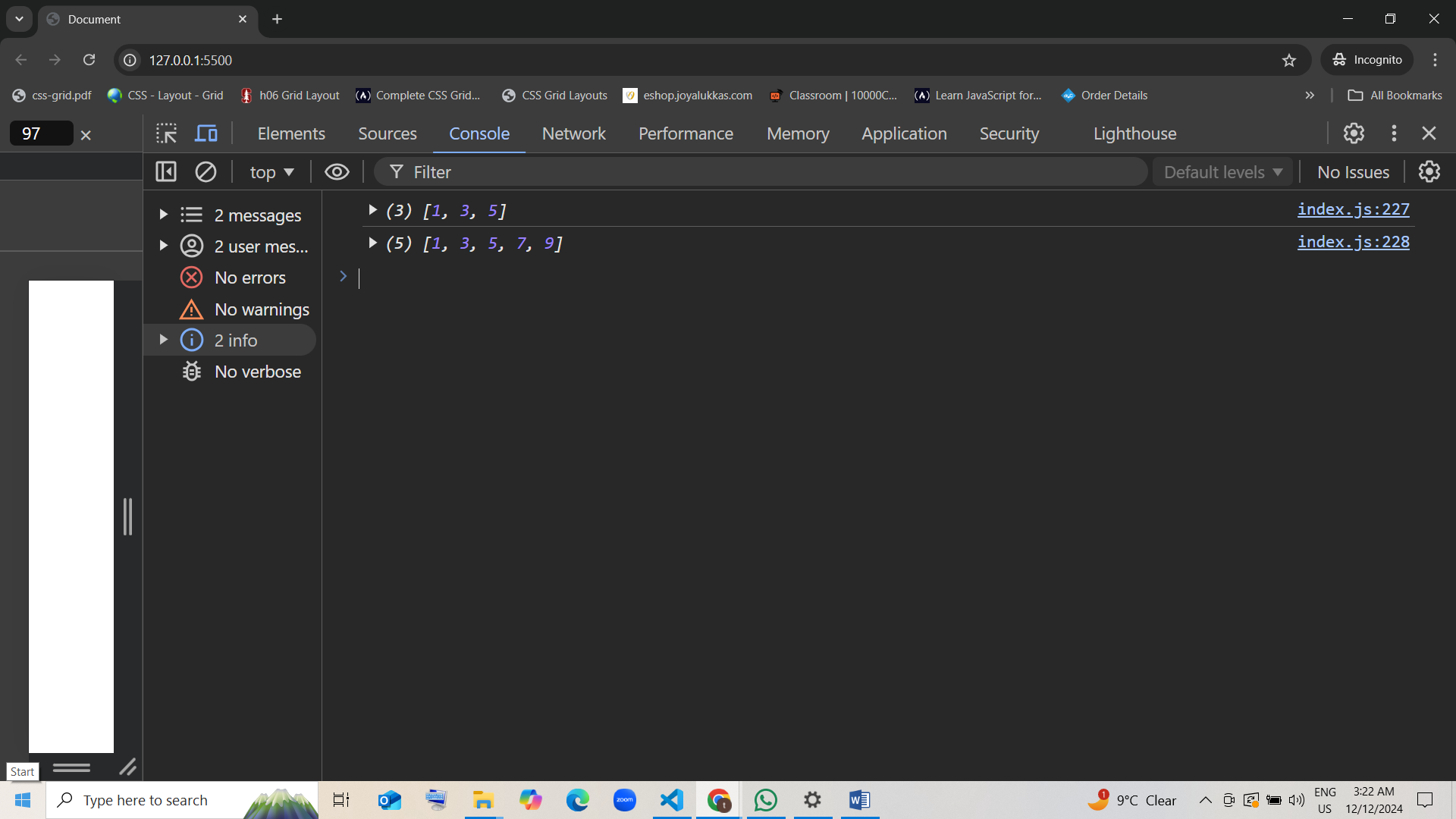
1. **Function with Return Inside Loop**

* function findFirstEven(numbers) {  
   for (let i = 0; i < numbers.length; i++) {  
   if (numbers[i] % 2 === 0) {  
   return numbers[i];  
   }  
   }  
   return null;  
  }  
    
  console.log(findFirstEven([1, 3, 5, 8]));  
  console.log(findFirstEven([1, 3, 5]));
* What will be the output of the code above?
  1. 8 and null guess [correct]
  2. 1 and null
  3. null and 5
  4. 8 and 5



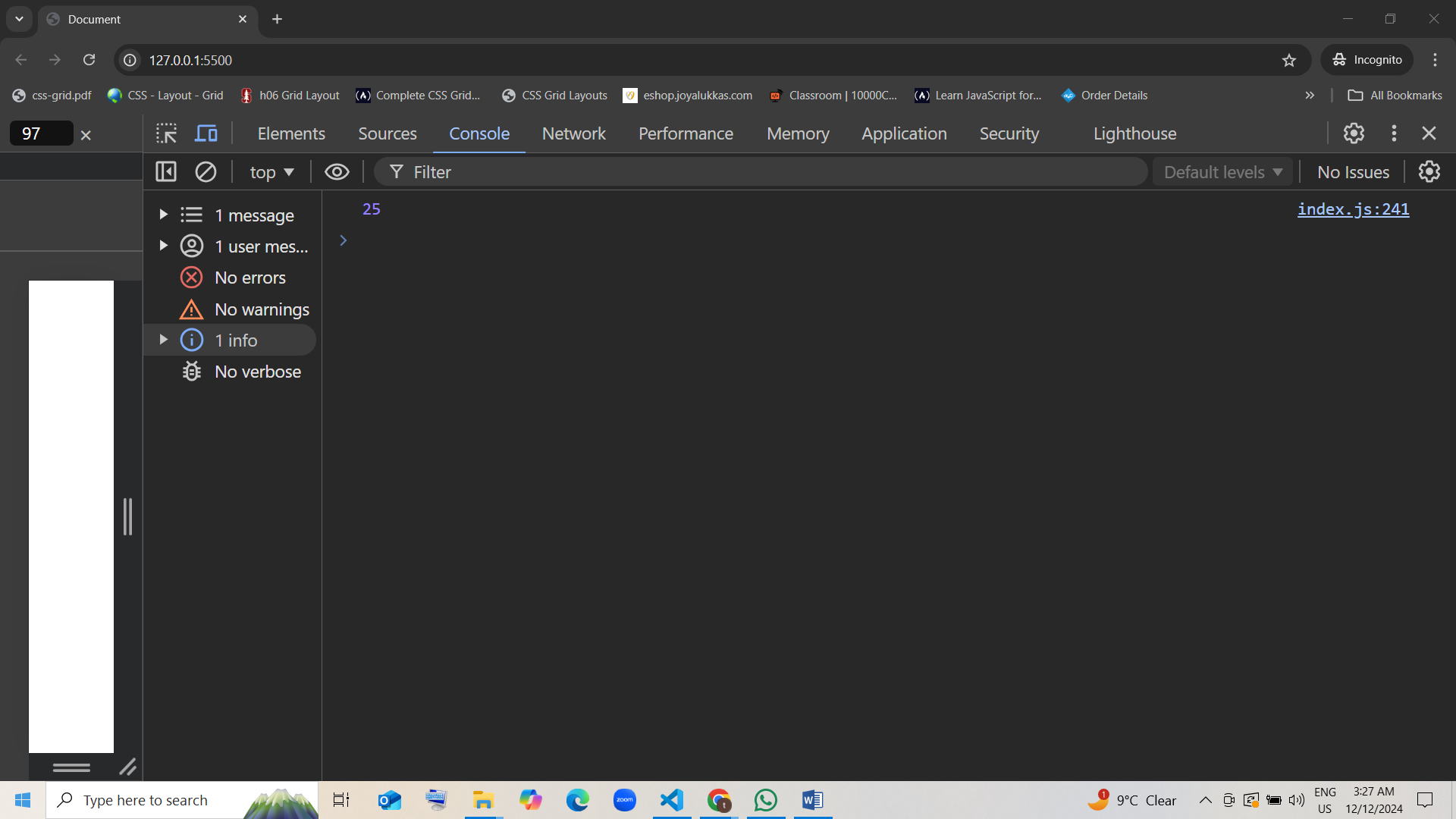
1. **Function Returning an Array**

* function getOddNumbers(n) {  
   let odds = [];  
   for (let i = 1; i <= n; i++) {  
   if (i % 2 !== 0) {  
   odds.push(i);  
   }  
   }  
   return odds;  
  }  
    
  console.log(getOddNumbers(5));  
  console.log(getOddNumbers(10));
* What will be the output of the code above?
  1. [1, 3, 5] and [1, 3, 5, 7, 9] guess [correct]
  2. [1, 3] and [1, 3, 5, 7]
  3. [5, 3, 1] and [9, 7, 5, 3]
  4. null and null



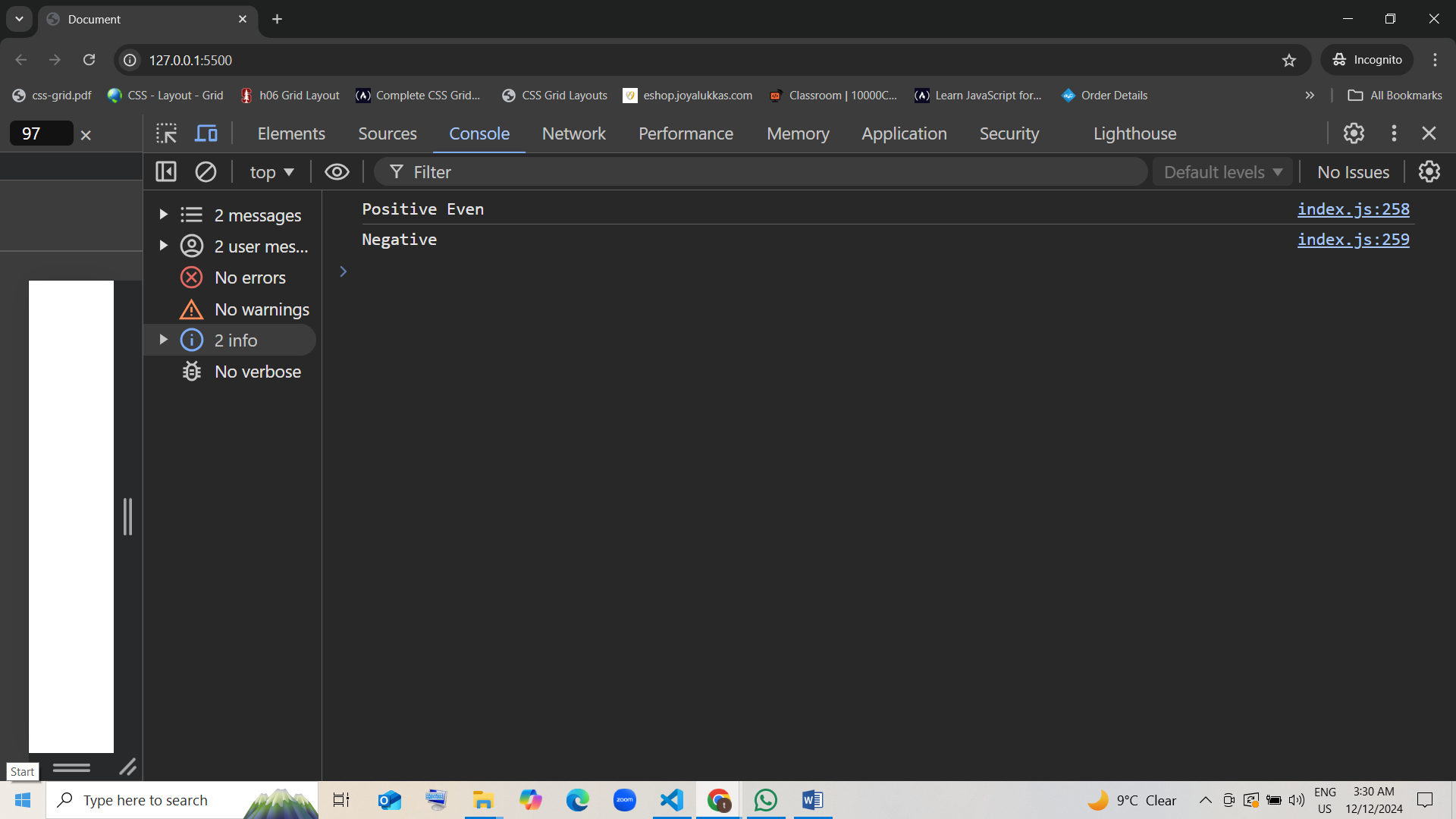
1. **Calling Function in Another Function**

* function square(num) {  
   return num \* num;  
  }  
    
  function sumOfSquares(a, b) {  
   return square(a) + square(b);  
  }  
    
  console.log(sumOfSquares(3, 4));
* What will be the output of the code above?
  1. 25 guess [correct]
  2. 49
  3. 7
  4. 25 and 49



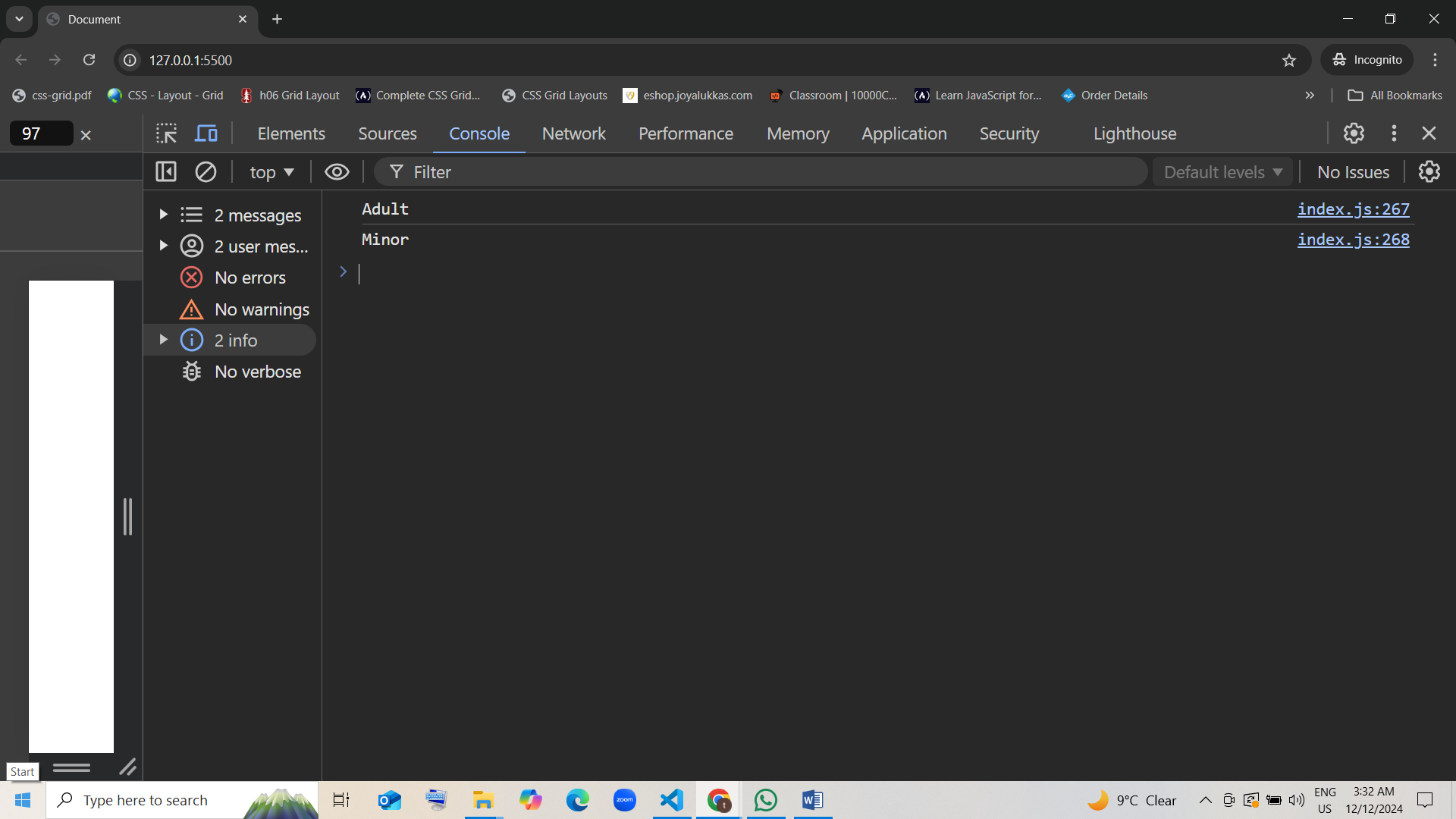
1. **Function with Nested Conditions**

* function numberCategory(num) {  
   if (num > 0) {  
   if (num % 2 === 0) {  
   return "Positive Even";  
   } else {  
   return "Positive Odd";  
   }  
   } else if (num < 0) {  
   return "Negative";  
   } else {  
   return "Zero";  
   }  
  }  
    
  console.log(numberCategory(2));  
  console.log(numberCategory(-5));
* What will be the output of the code above?
  1. Positive Even and Negative guess [correct]
  2. Positive Odd and Zero
  3. Negative and Positive Even
  4. Zero and Positive Even



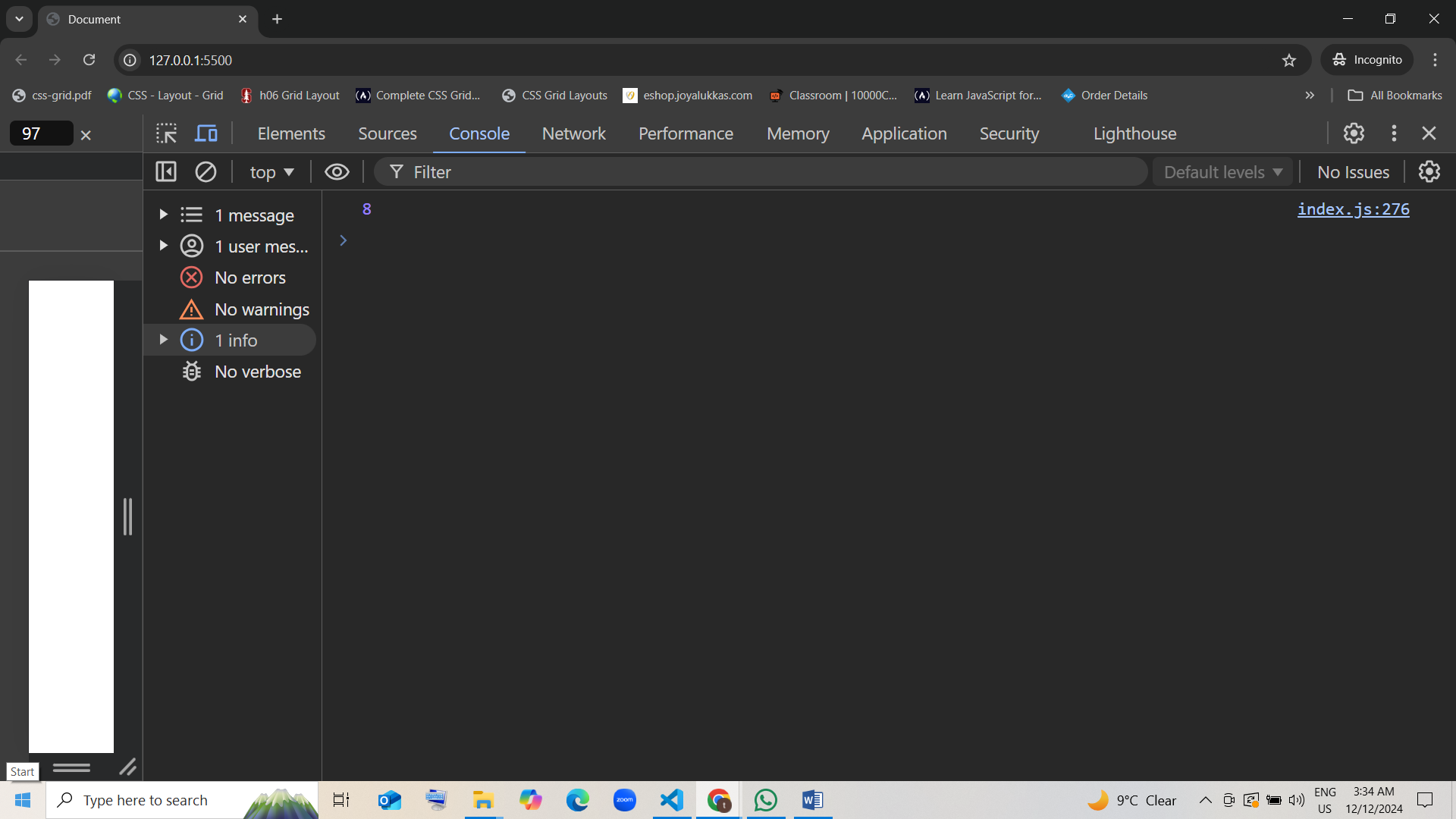
1. **Function with Ternary Operator**

* function isAdult(age) {  
   return age >= 18 ? "Adult" : "Minor";  
  }  
    
  console.log(isAdult(20));  
  console.log(isAdult(15));
* What will be the output of the code above?
  1. Adult and Minor guess [correct]
  2. Minor and Adult
  3. 20 and 15
  4. true and false



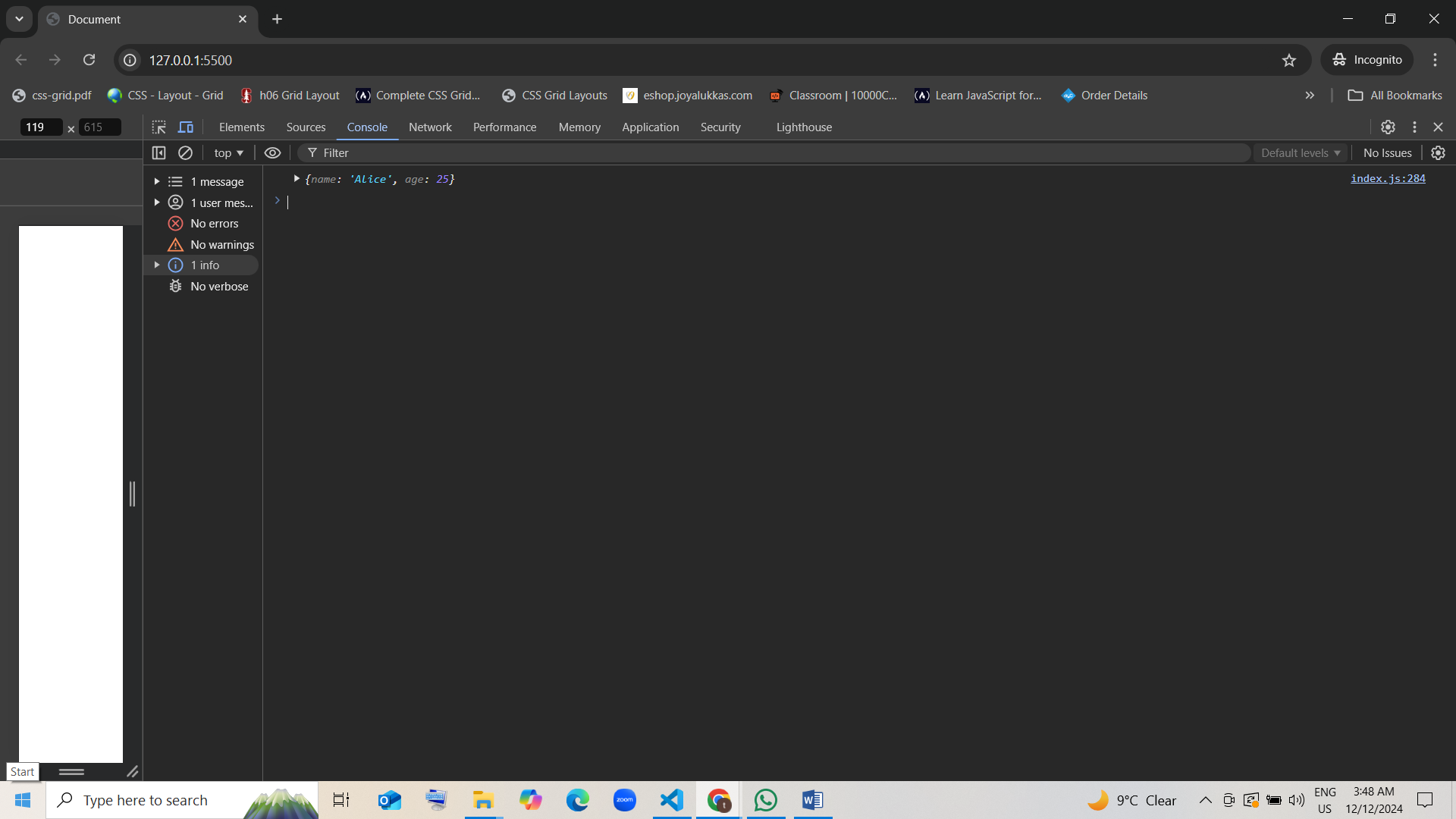
1. **Calling a Function Twice**

* function double(x) {  
   return x \* 2;  
  }  
    
  console.log(double(double(2))); // Nested function call (double function)
* What will be the output of the code above?
  1. 8 [correct]
  2. 4
  3. 16
  4. Error



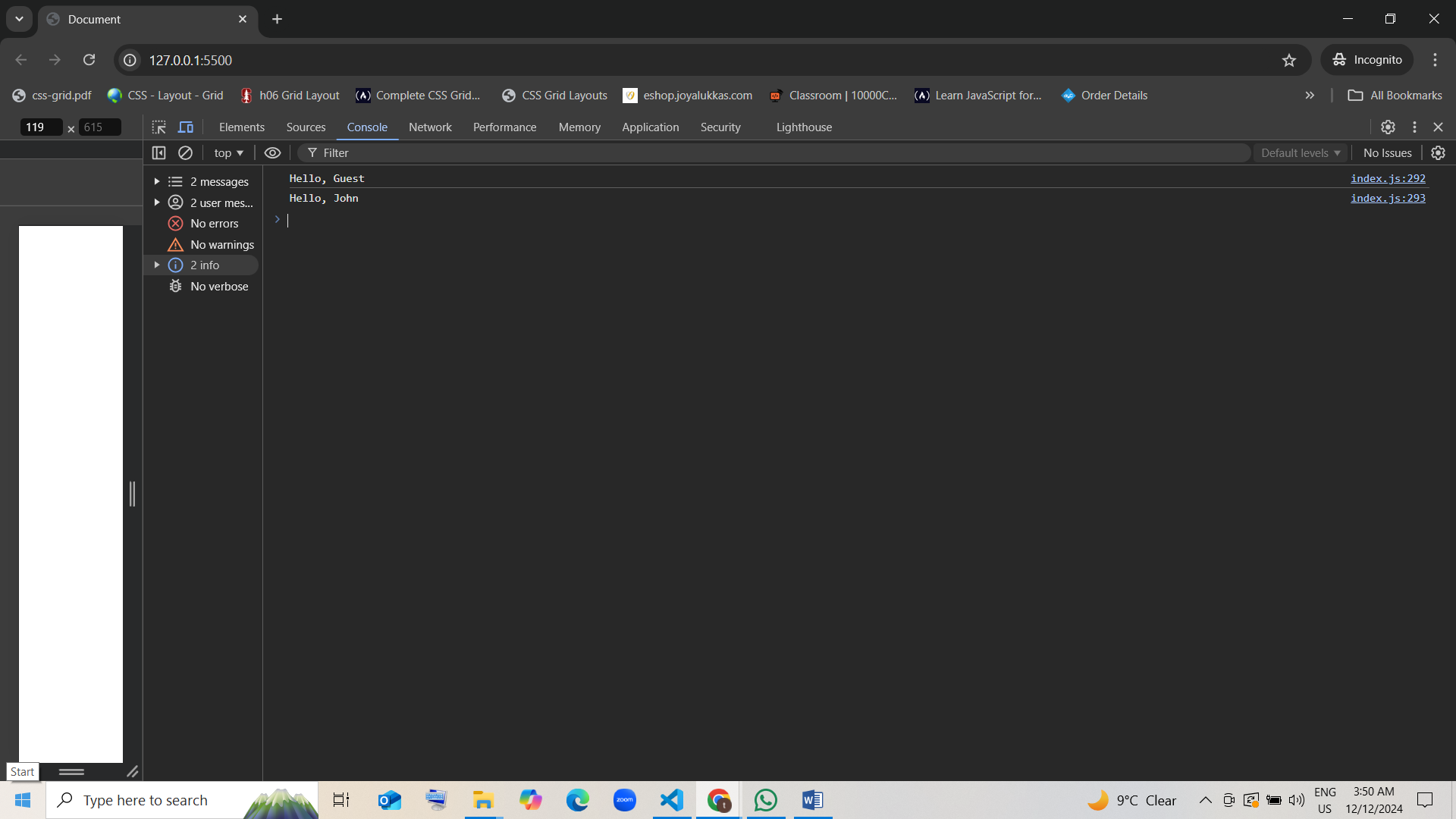
1. **Return Object from Function**

* function createPerson(name, age) {  
   return {name: name, age: age};  
  }  
    
  console.log(createPerson("Alice", 25));
* What will be the output of the code above?
  1. {name: "Alice", age: 25} guess [correct]
  2. ["Alice", 25]
  3. {"name": "Alice", "age": 25}
  4. 25, Alice



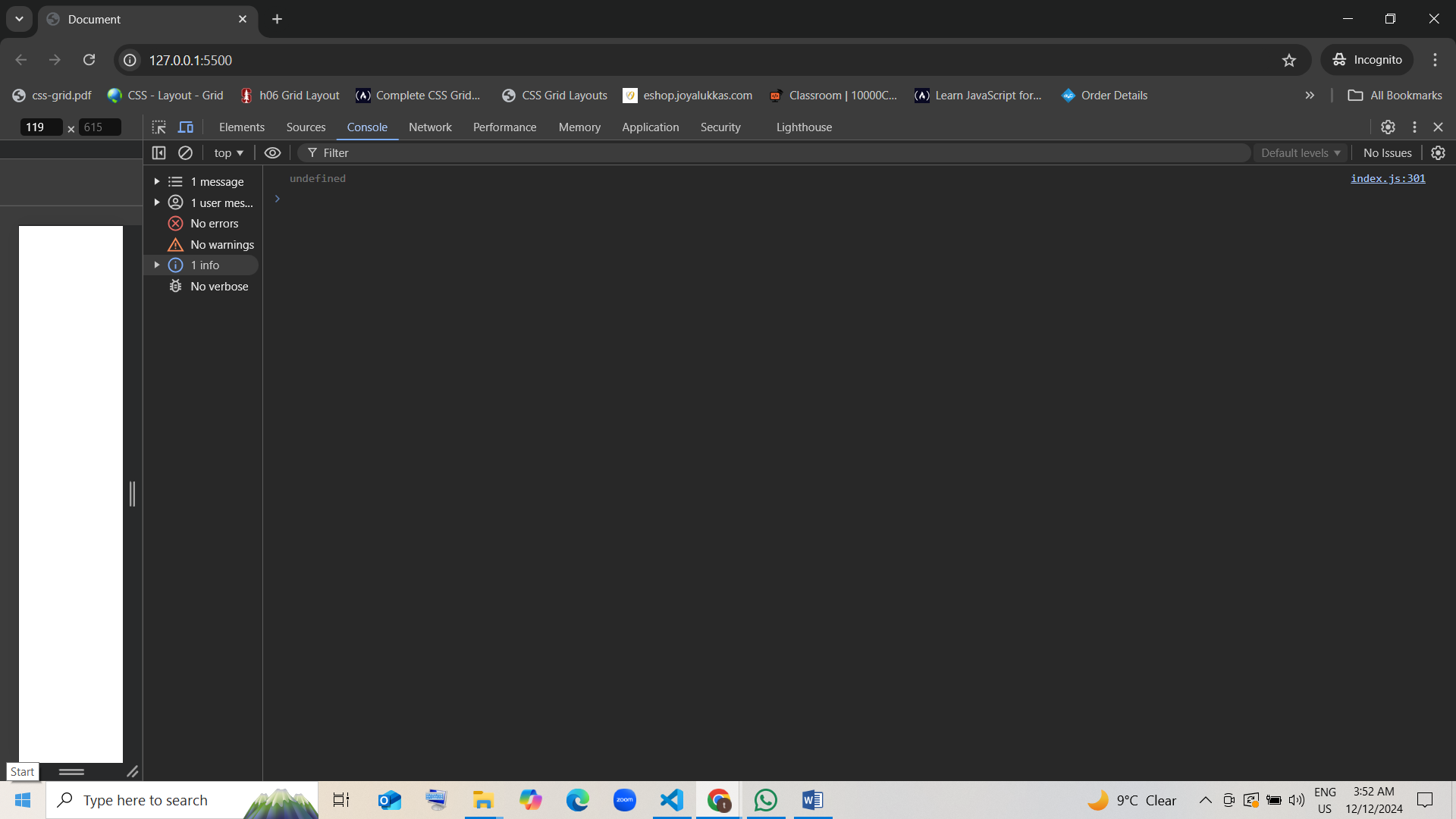
1. **Calling Function with Default Parameters**

* function greet(name = "Guest") {  
   return "Hello, " + name;  
  }  
    
  console.log(greet());  
  console.log(greet("John"));
* What will be the output of the code above?
  1. Hello, Guest and Hello, John guess [correct]
  2. Hello, John and Hello, Guest
  3. undefined and "John"
  4. null and "Guest"



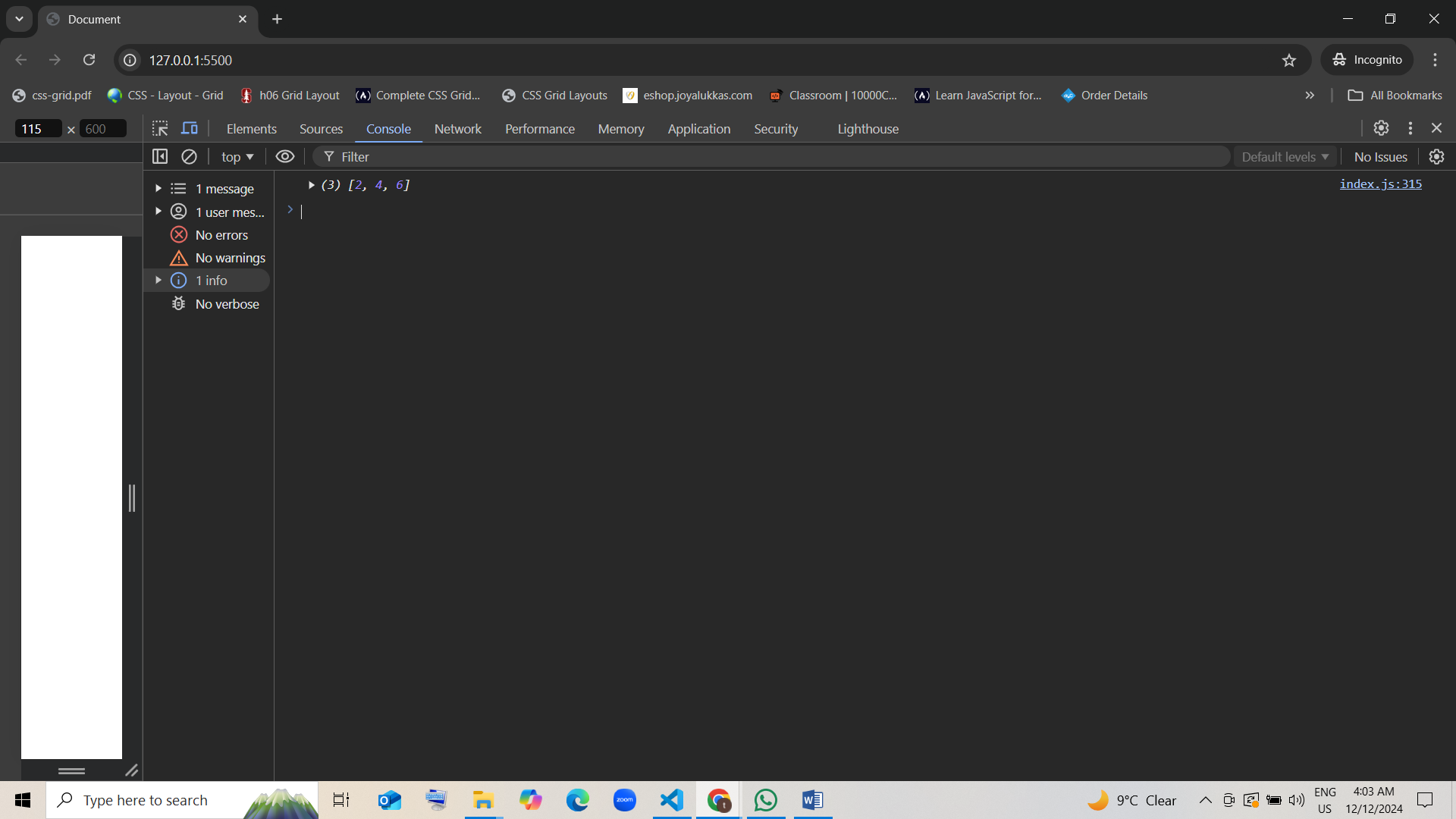
1. **Returning Undefined Explicitly**

* function doNothing() {  
   return;  
  }  
    
  console.log(doNothing());
* What will be the output of the code above?
  1. Undefined guess[correct]
  2. null
  3. 0
  4. Error



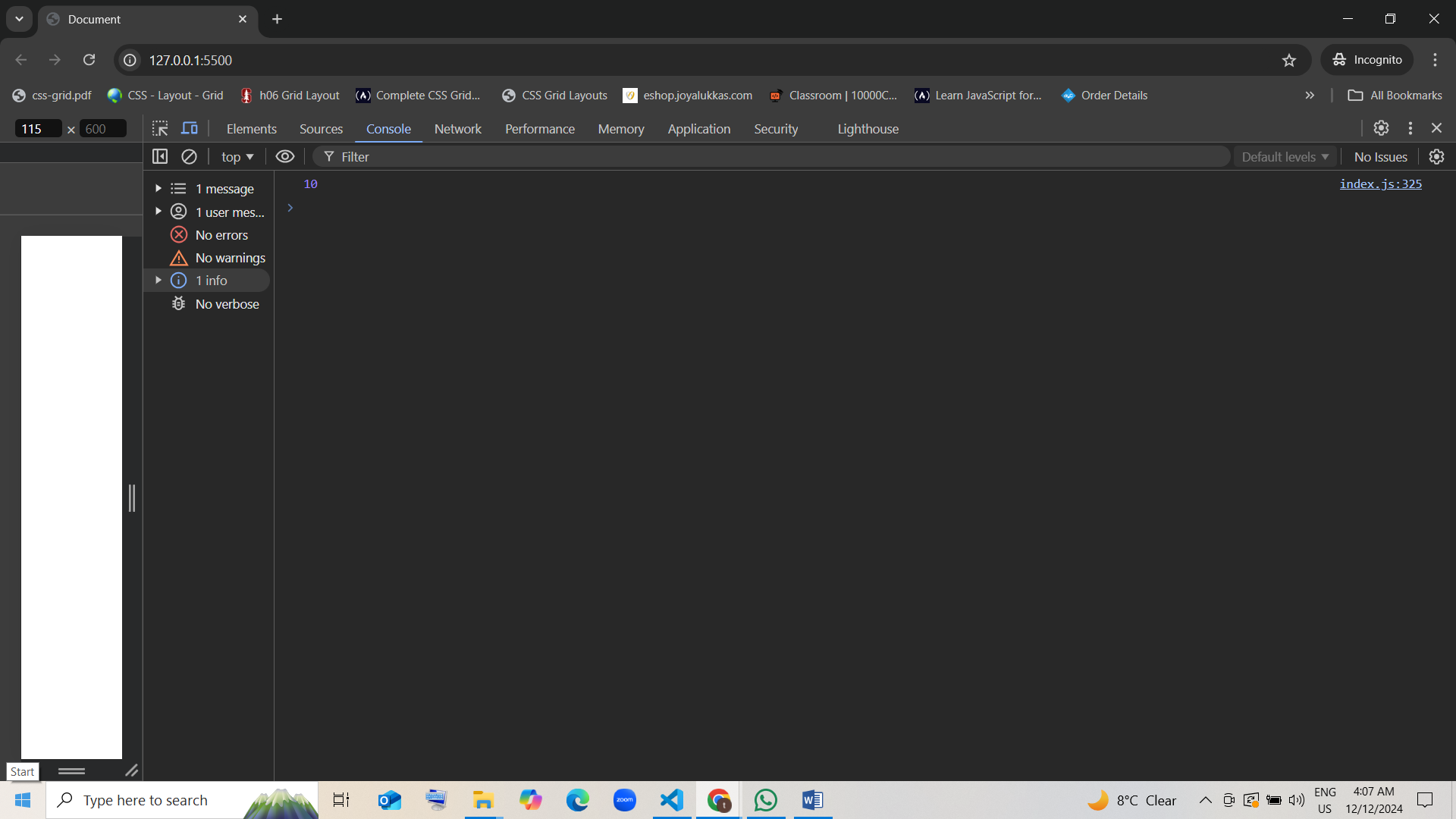
1. **Loop with Continue Statement**

* function getEvenNumbers(n) {  
   let evens = [];  
   for (let i = 1; i <= n; i++) {  
   if (i % 2 !== 0) {  
   continue;  
   }  
   evens.push(i);  
   }  
   return evens;  
  }  
    
  console.log(getEvenNumbers(6));
* What will be the output of the code above?
  1. [2, 4, 6] [correct]
  2. [1, 3, 5]
  3. [6, 4, 2]
  4. [6]



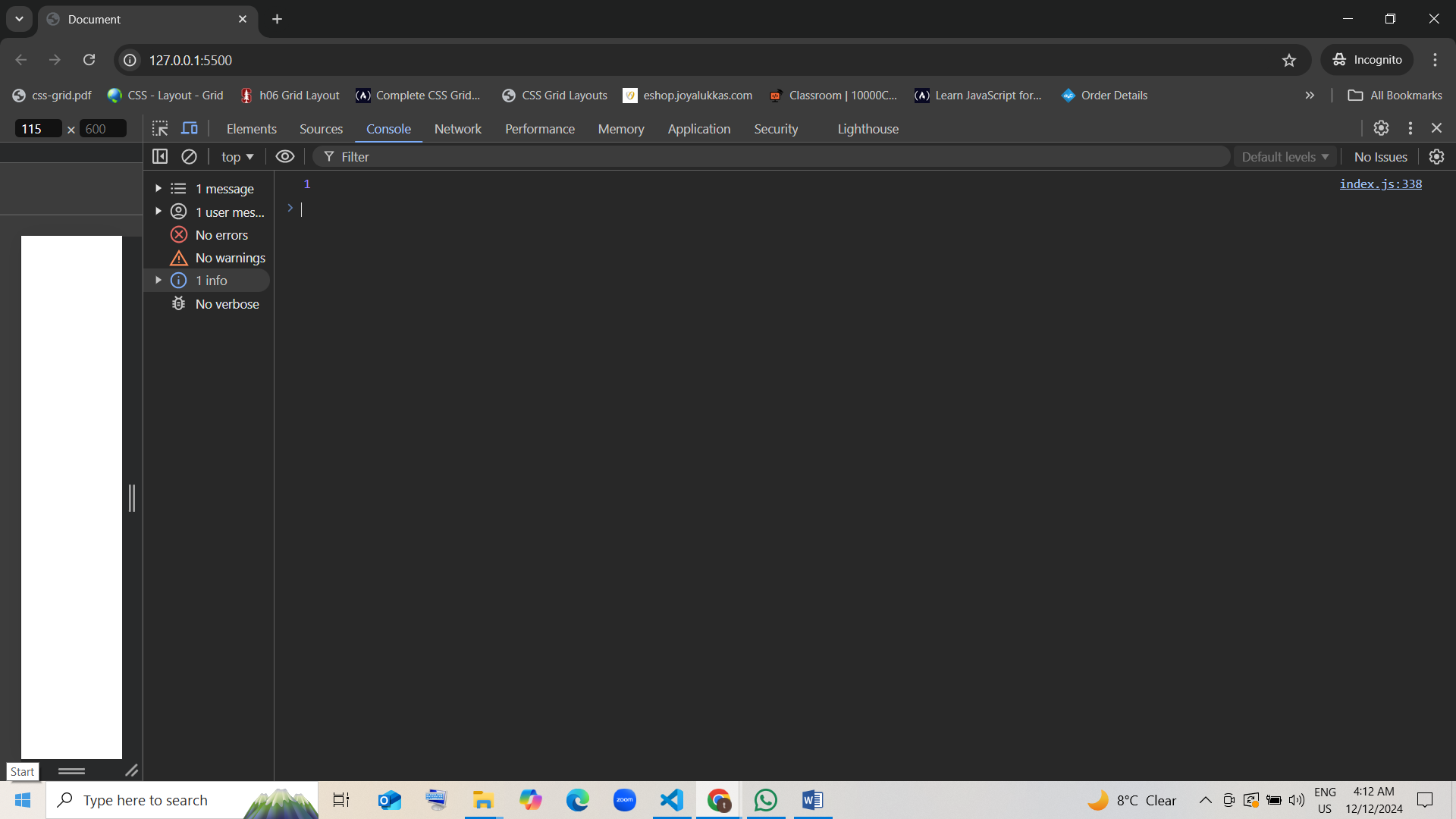
1. **Function Inside an Object**

* let calculator = {  
   add: function(a, b) {   
   return a + b;  
   }  
  };  
    
  console.log(calculator.add(4, 6));
* What will be the output of the code above?
  1. 10 guess [correct]
  2. 46
  3. Error
  4. Undefined



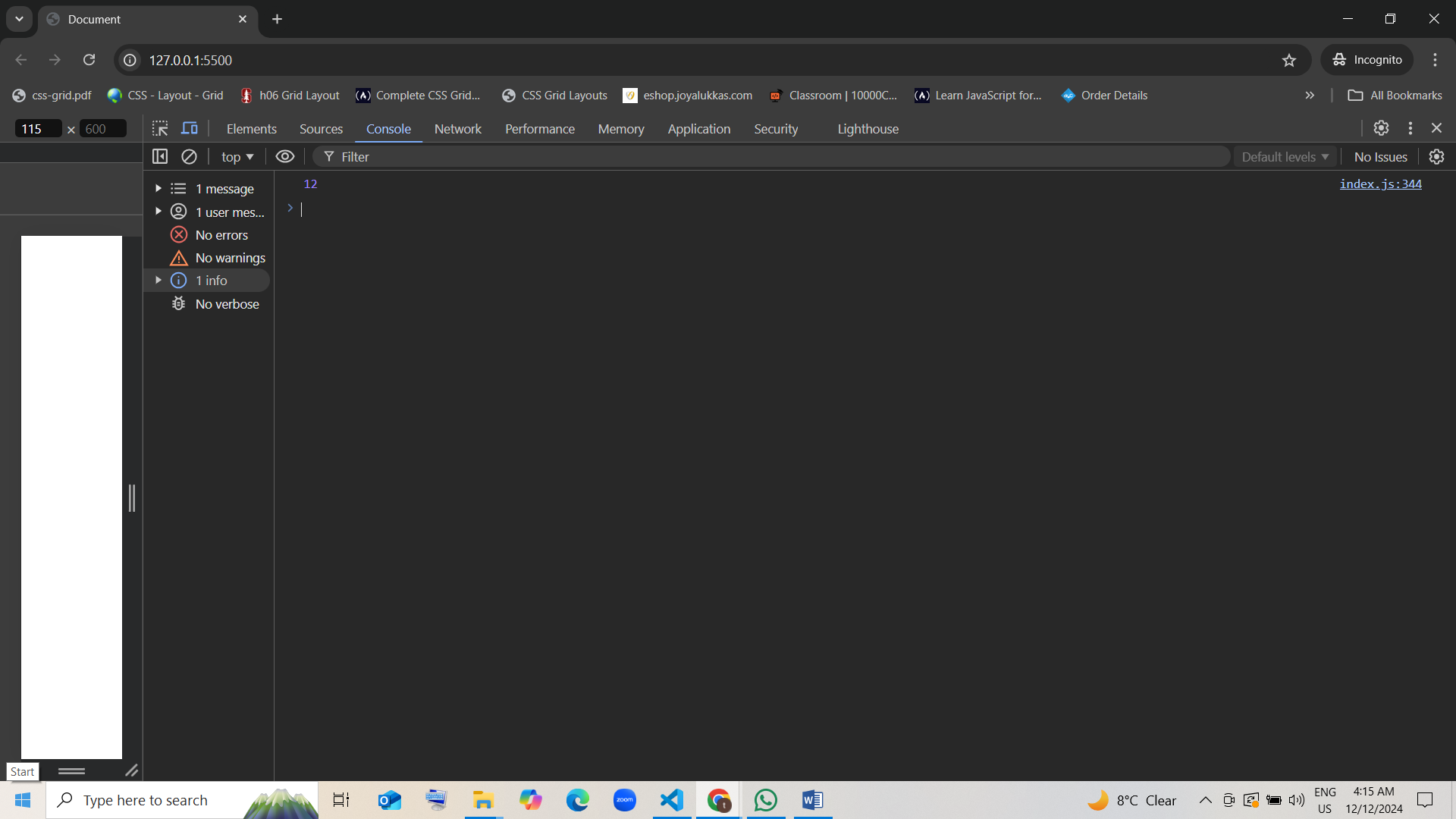
1. **Return in For Loop**

* function getFirstPositive(numbers) {  
   for (let i = 0; i < numbers.length; i++) {  
   if (numbers[i] > 0) {  
   return numbers[i];  
   }  
   }  
   return -1;  
  }  
    
  console.log(getFirstPositive([-3, -2, 1, 2]));
* What will be the output of the code above?
  1. 1 guess [correct]
  2. -1
  3. 2
  4. 0



1. **Return Value from Arrow Function**

* let multiply = (a, b) => a \* b;  
    
  console.log(multiply(3, 4));
* What will be the output of the code above?
  1. 12 guess [correct]
  2. 34
  3. Error
  4. 7



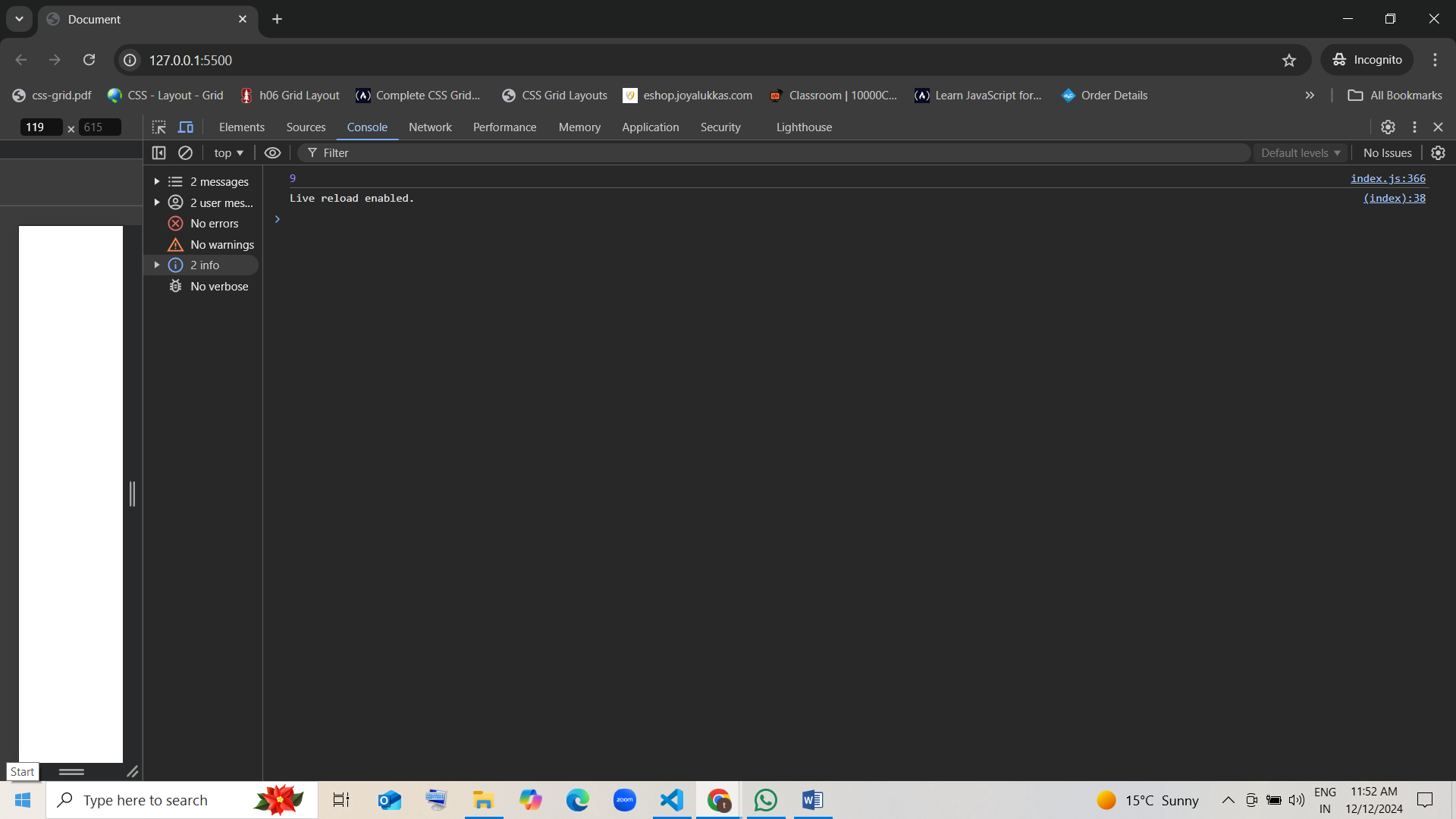
1. **Function with No Return**

* function logMessage() {  
   console.log("Hello!");  
  }  
    
  logMessage();
* What will be the output of the code above?
  1. Hello! Guess [correct]
  2. undefined
  3. null
  4. Error



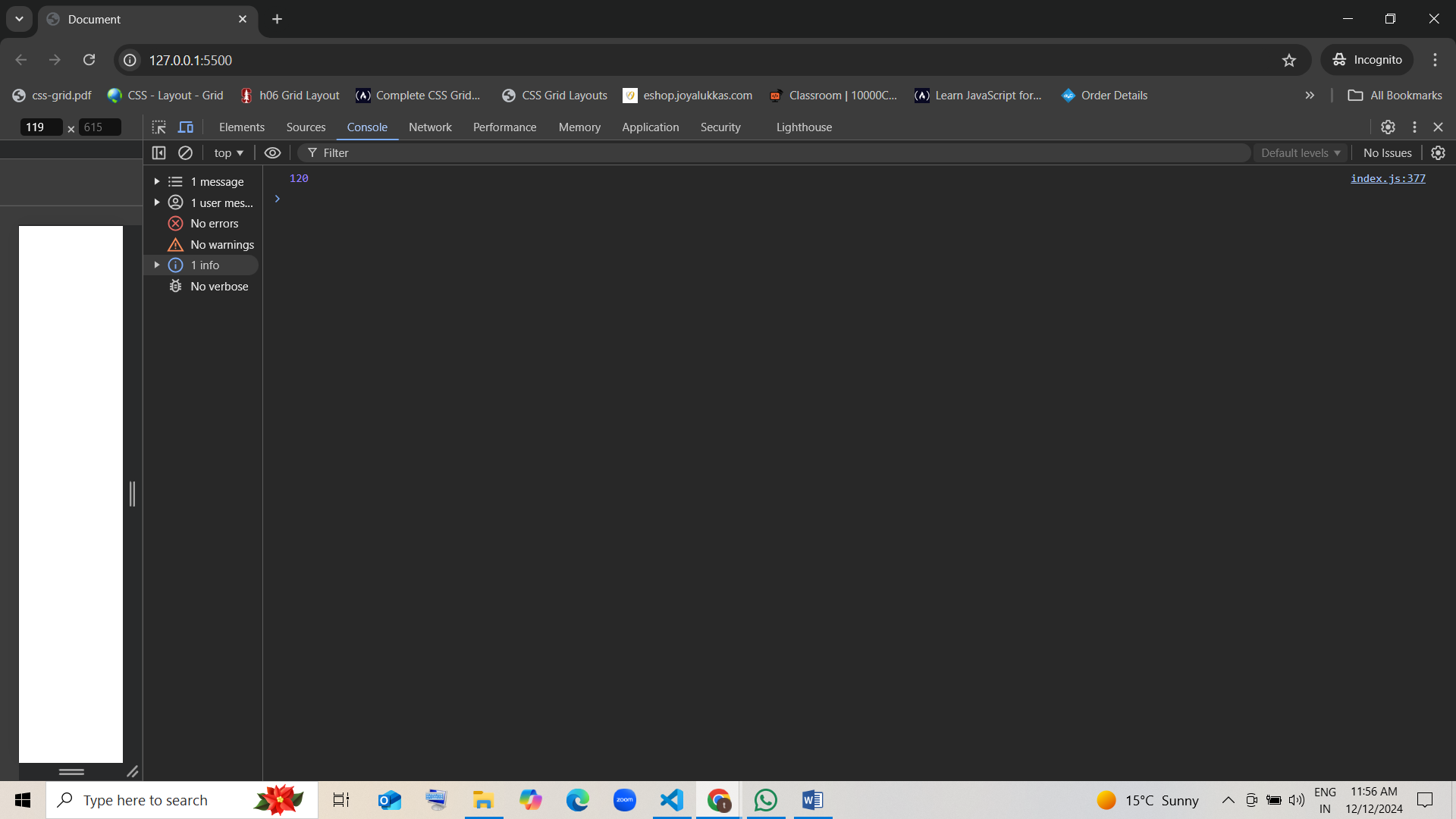
1. **Return in While Loop**

* function findFirstDivisibleBy3(numbers) {  
   let i = 0;  
   while (i < numbers.length) {  
   if (numbers[i] % 3 === 0) {  
   return numbers[i];  
   }  
   i++;  
   }  
   return -1;  
  }  
    
  console.log(findFirstDivisibleBy3([1, 2, 9, 4]));
* What will be the output of the code above?
  1. 9 guess [correct]
  2. -1
  3. 4
  4. 1



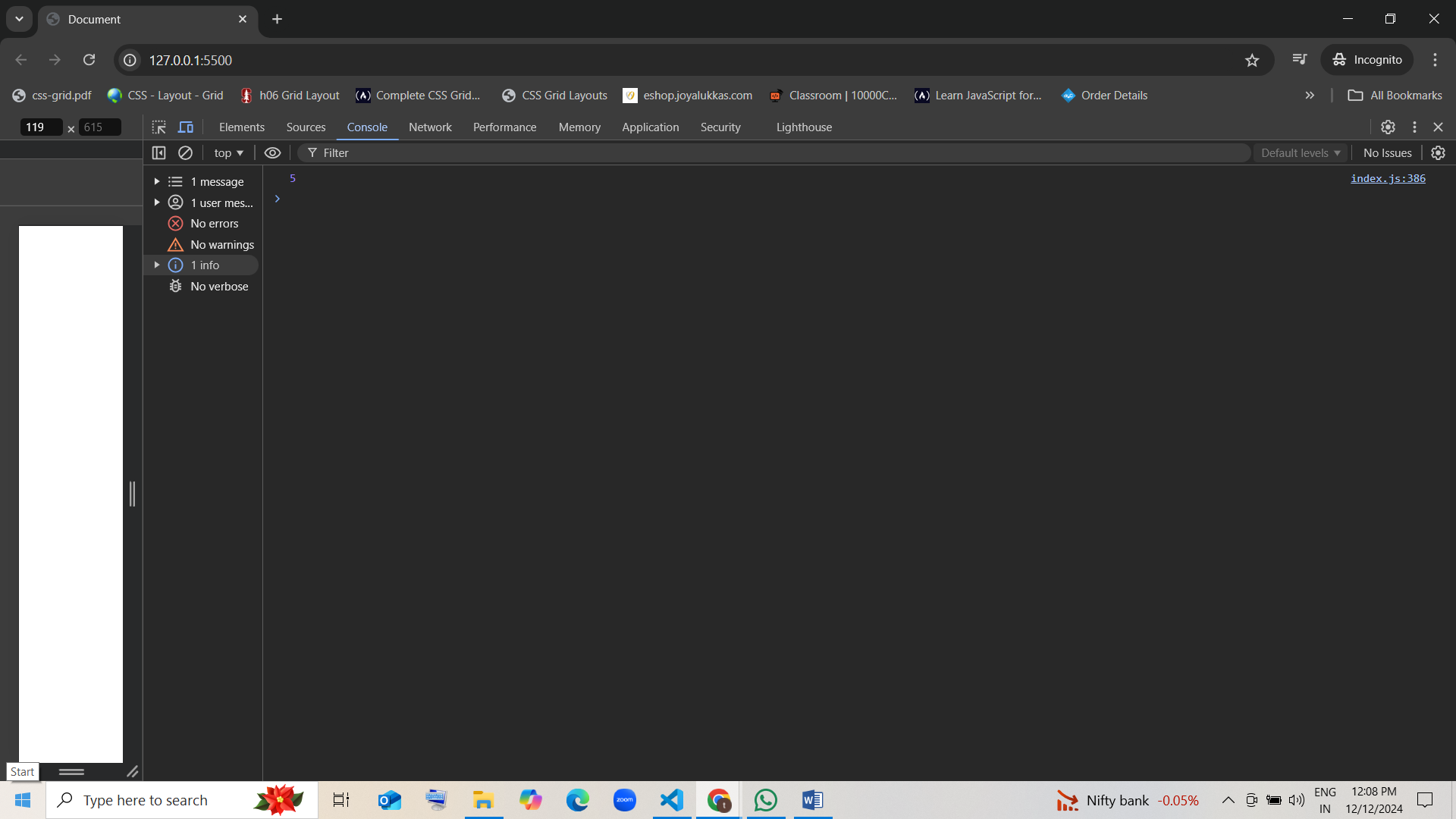
1. **Recursive Function Call**

* function factorial(n) {  
   if (n === 0) {  
   return 1;  
   }  
   return n \* factorial(n - 1);  
  }  
    
  console.log(factorial(5));
* What will be the output of the code above?
  1. 120 [correct] [5!= 5\*4\*3\*2\*1=120]
  2. 25
  3. 1
  4. 5



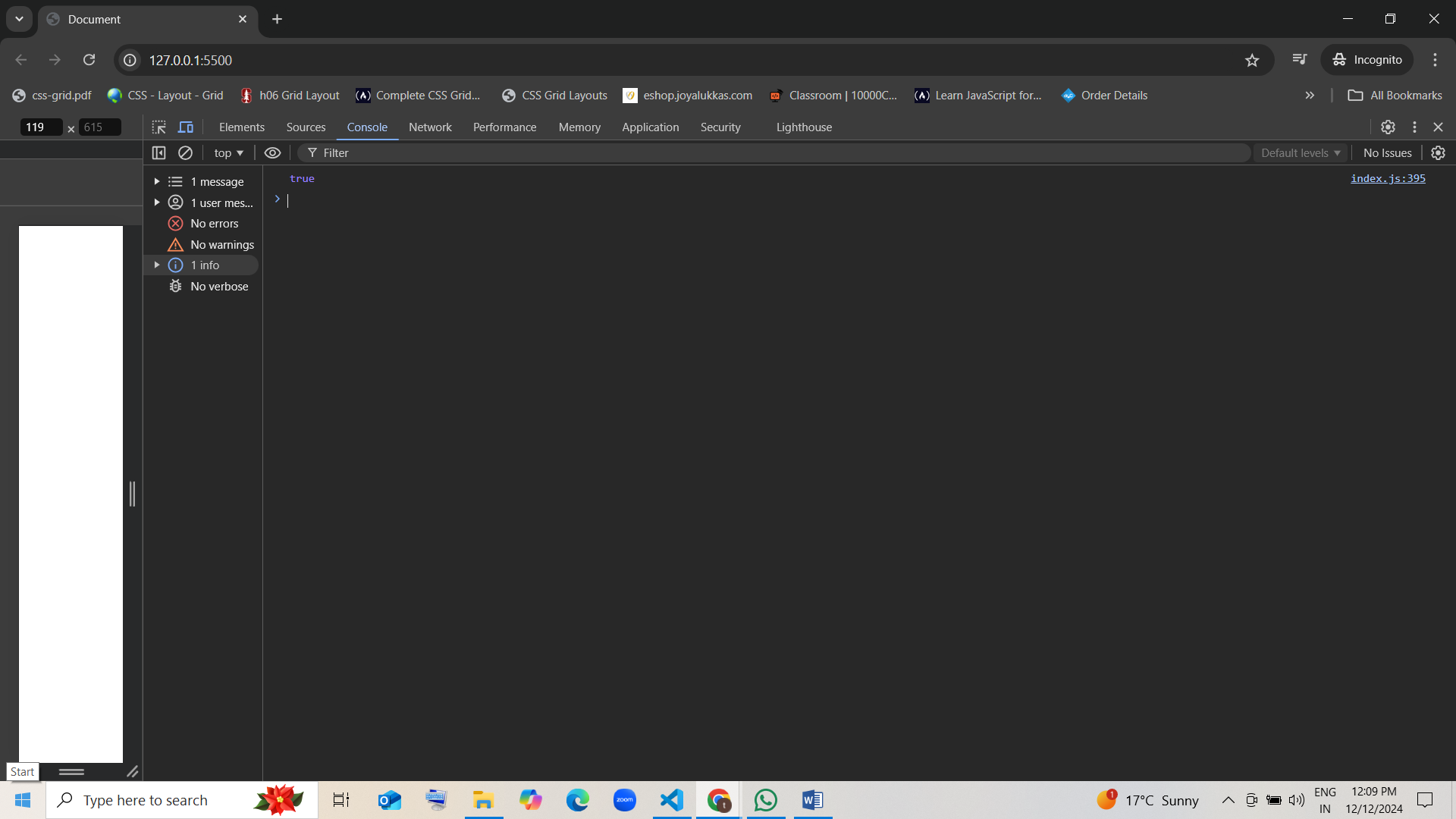
1. **Using Function Expression**

* let divide = function(a, b) {  
   return a / b;  
  };  
    
  console.log(divide(10, 2));
* What will be the output of the code above?
  1. 5 guess [correct]
  2. 10
  3. 2
  4. Undefined



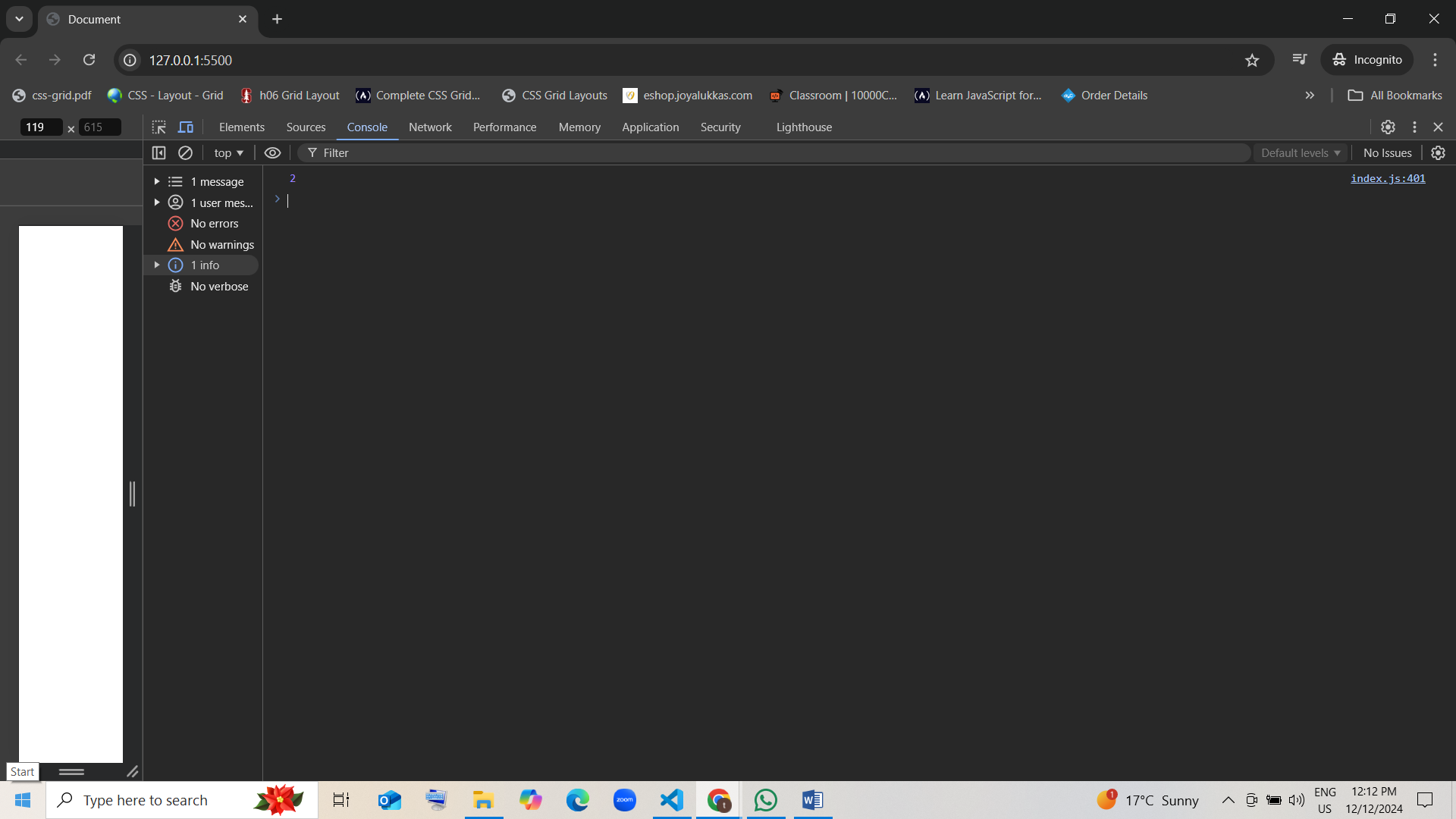
1. **Return Boolean Value**

* function isNegative(num) {  
   return num < 0;  
  }  
    
  console.log(isNegative(-5));
* What will be the output of the code above?
  1. True guess [correct]
  2. false
  3. null
  4. undefined



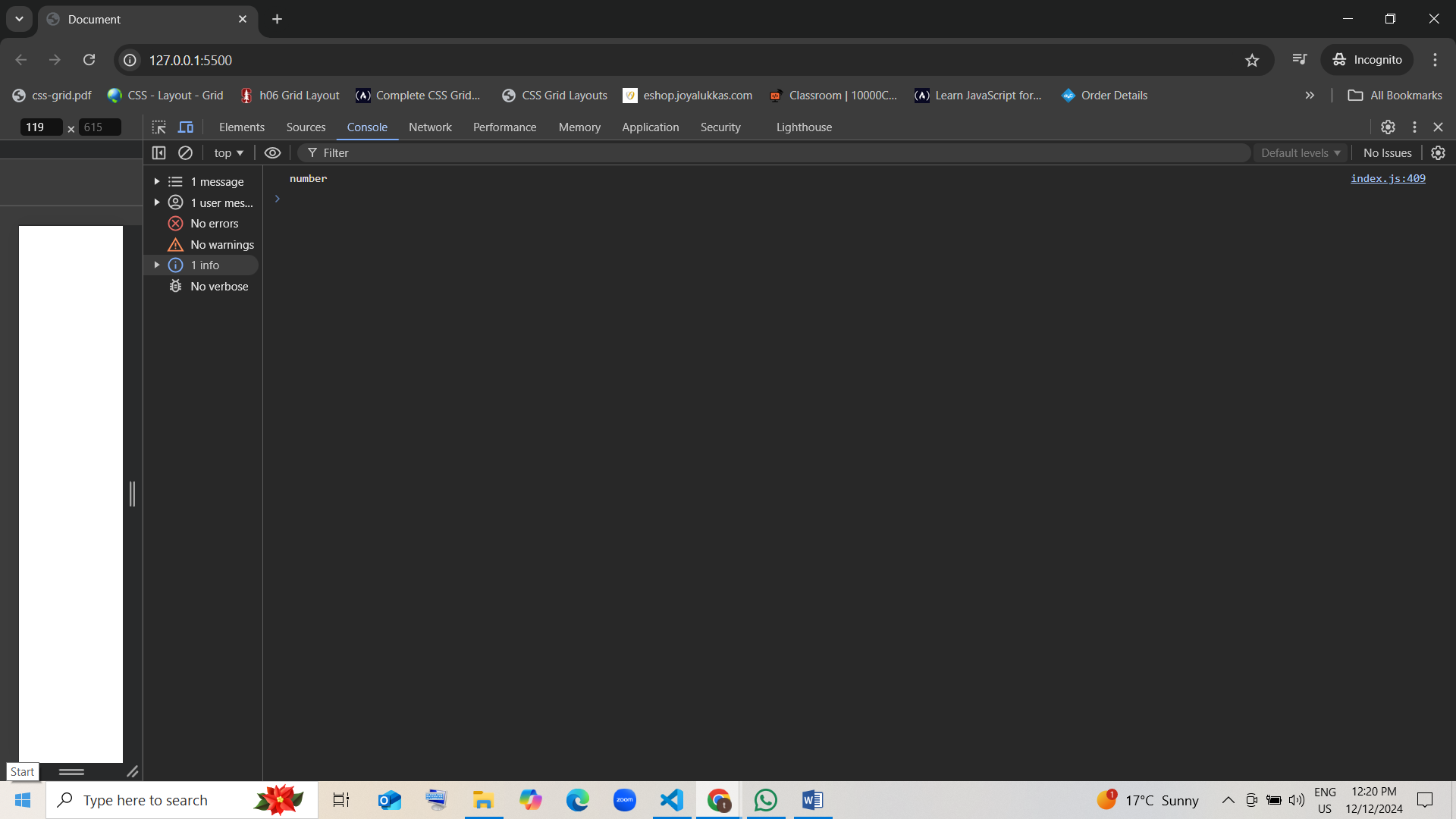
1. **Function Call Inside an Array**

* let array = [function() { return 1 }, function() { return 2 }];  
    
  console.log(array[1]());
* What will be the output of the code above?
  1. 2 [correct]
  2. 1
  3. undefined
  4. Error



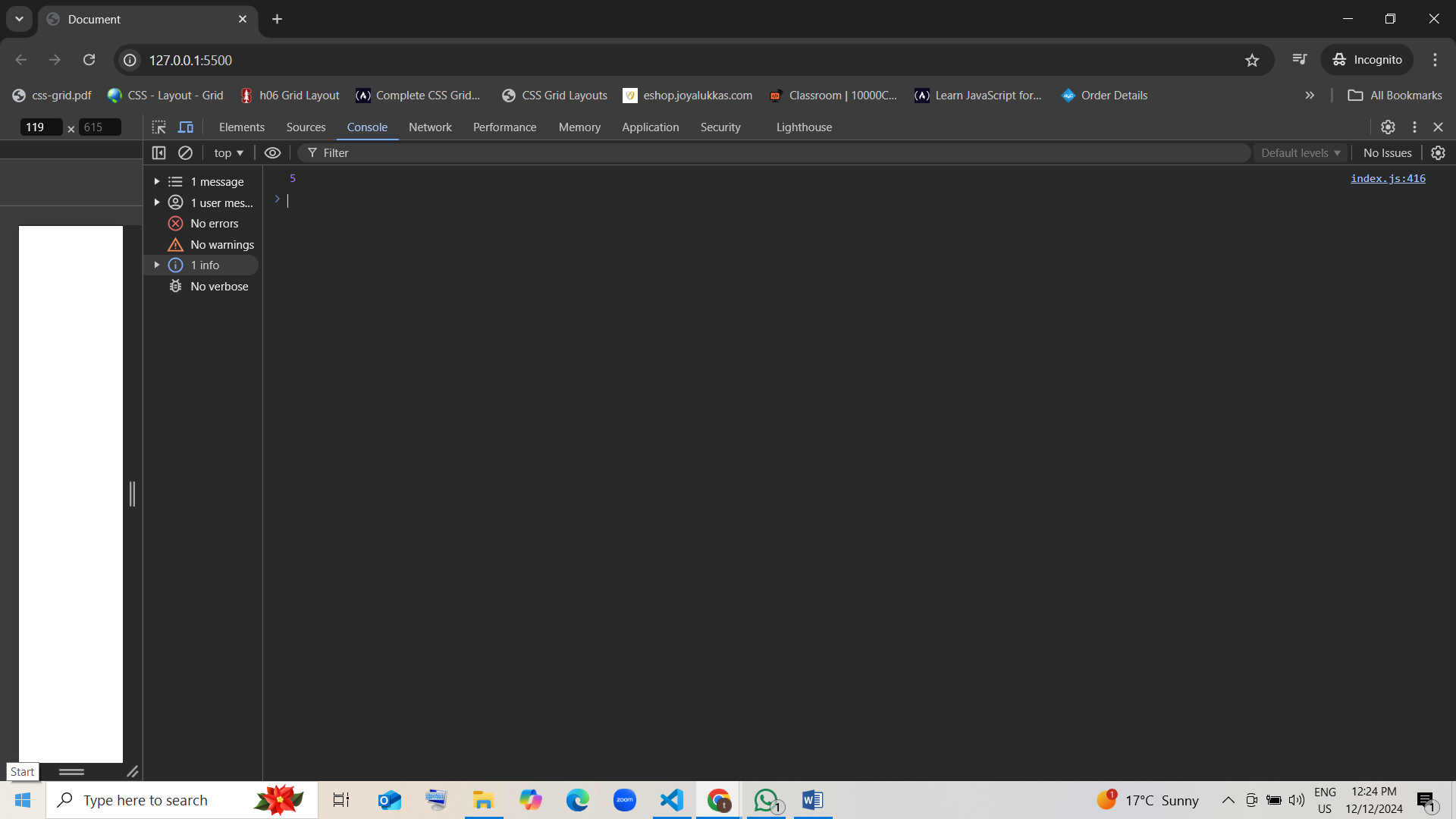
1. **Return Type of Function**

* function square(num) {  
   return num \* num;  
  }  
    
  console.log(typeof square(4));
* What will be the output of the code above?
  1. Number guess [correct]
  2. string
  3. undefined
  4. object



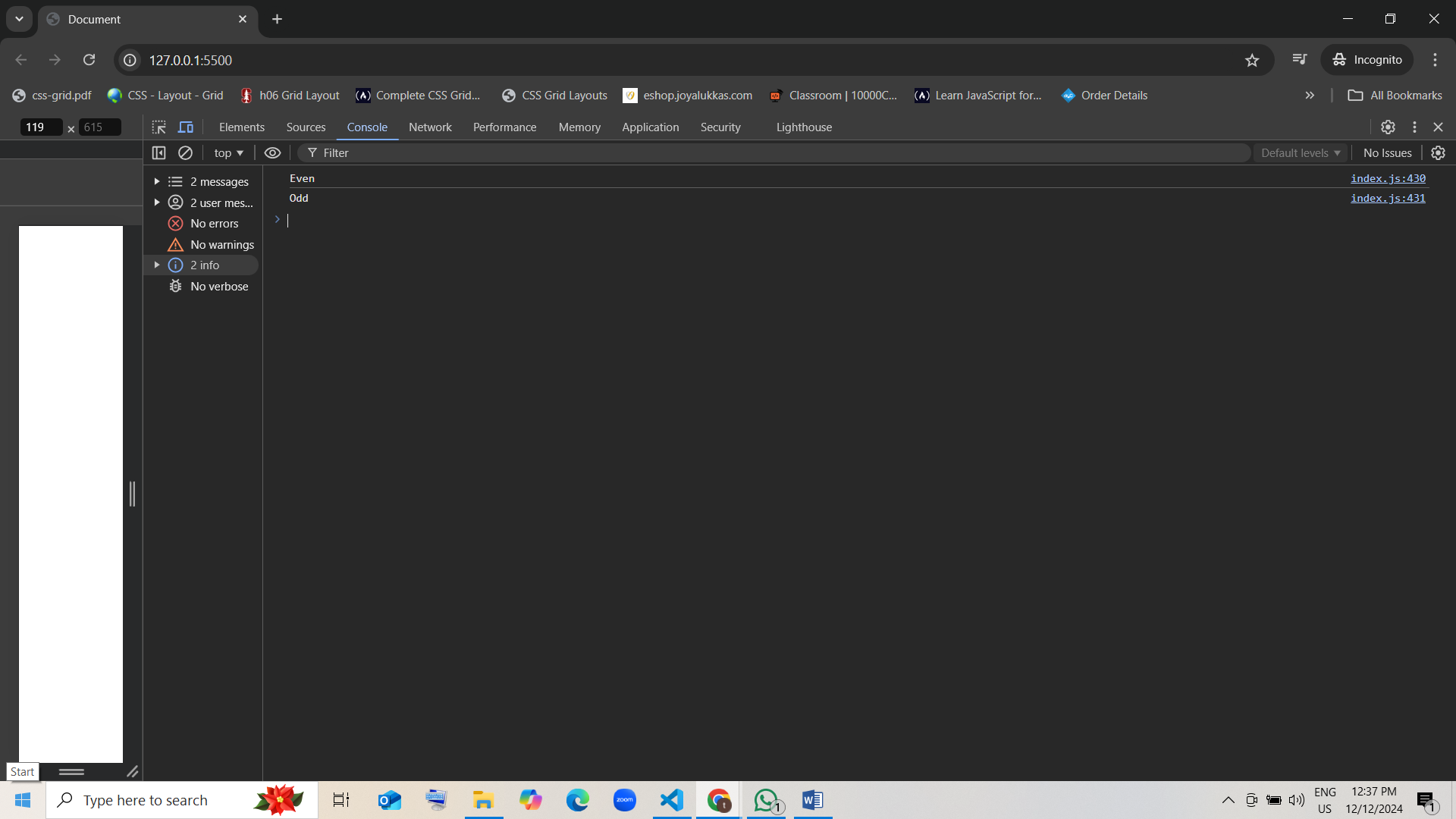
1. **Passing Function as Argument**

* function runOperation(operation, x, y) {  
   return operation(x, y);  
  }  
    
  console.log(runOperation((a, b) => a - b, 7, 2));
* What will be the output of the code above?
  1. 5 [correct]
  2. 9
  3. 2
  4. Error



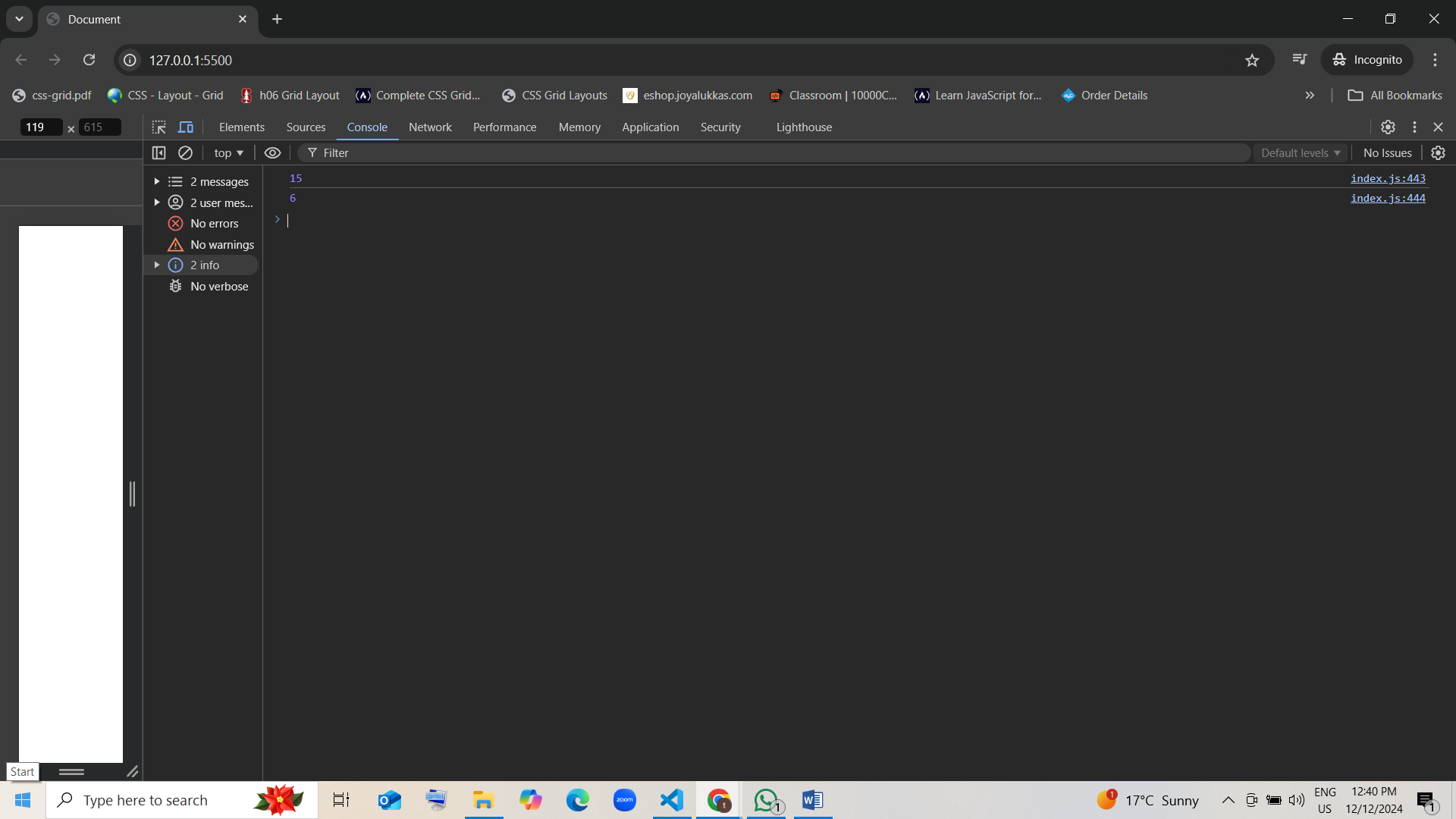
1. **Function with Return Type and Conditional Statement**

* function checkEvenOrOdd(number) {  
   if (number % 2 === 0) {  
   return "Even";  
   } else {  
   return "Odd";  
   }  
  }  
    
  console.log(checkEvenOrOdd(8));  
  console.log(checkEvenOrOdd(7));
* What will be the output of the code above?
  1. Even and Odd guess [correct]
  2. Odd and Even
  3. Odd and Odd
  4. Even and Even



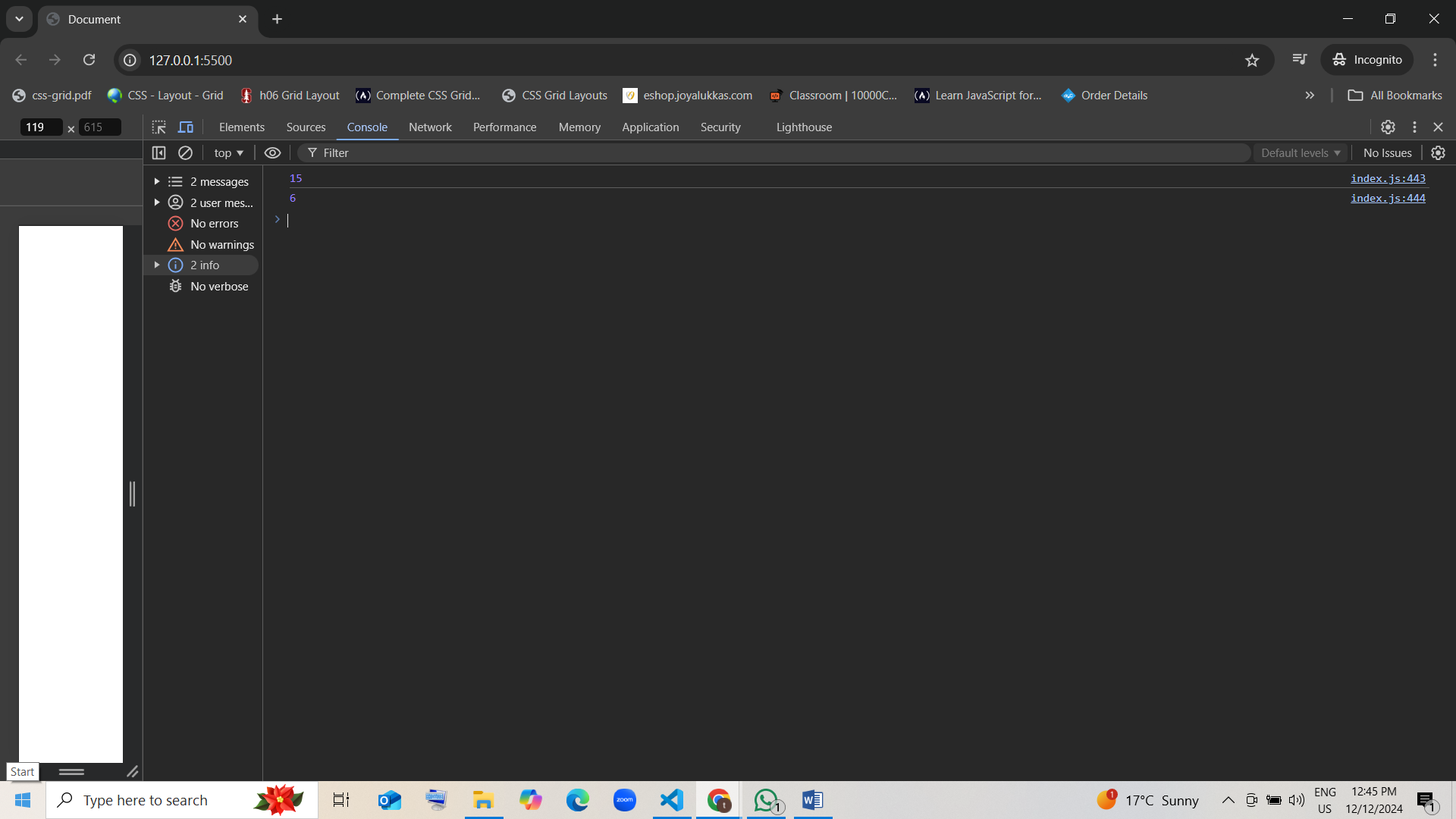
1. **Function with Loops**

* function sumNumbers(n) {  
   let sum = 0;  
   for (let i = 1; i <= n; i++) {  
   sum += i;  
   }  
   return sum;  
  }  
    
  console.log(sumNumbers(5));  
  console.log(sumNumbers(3));
* What will be the output of the code above?
  1. 15 and 6 [correct]
  2. 10 and 3
  3. 15 and 9
  4. 5 and 3



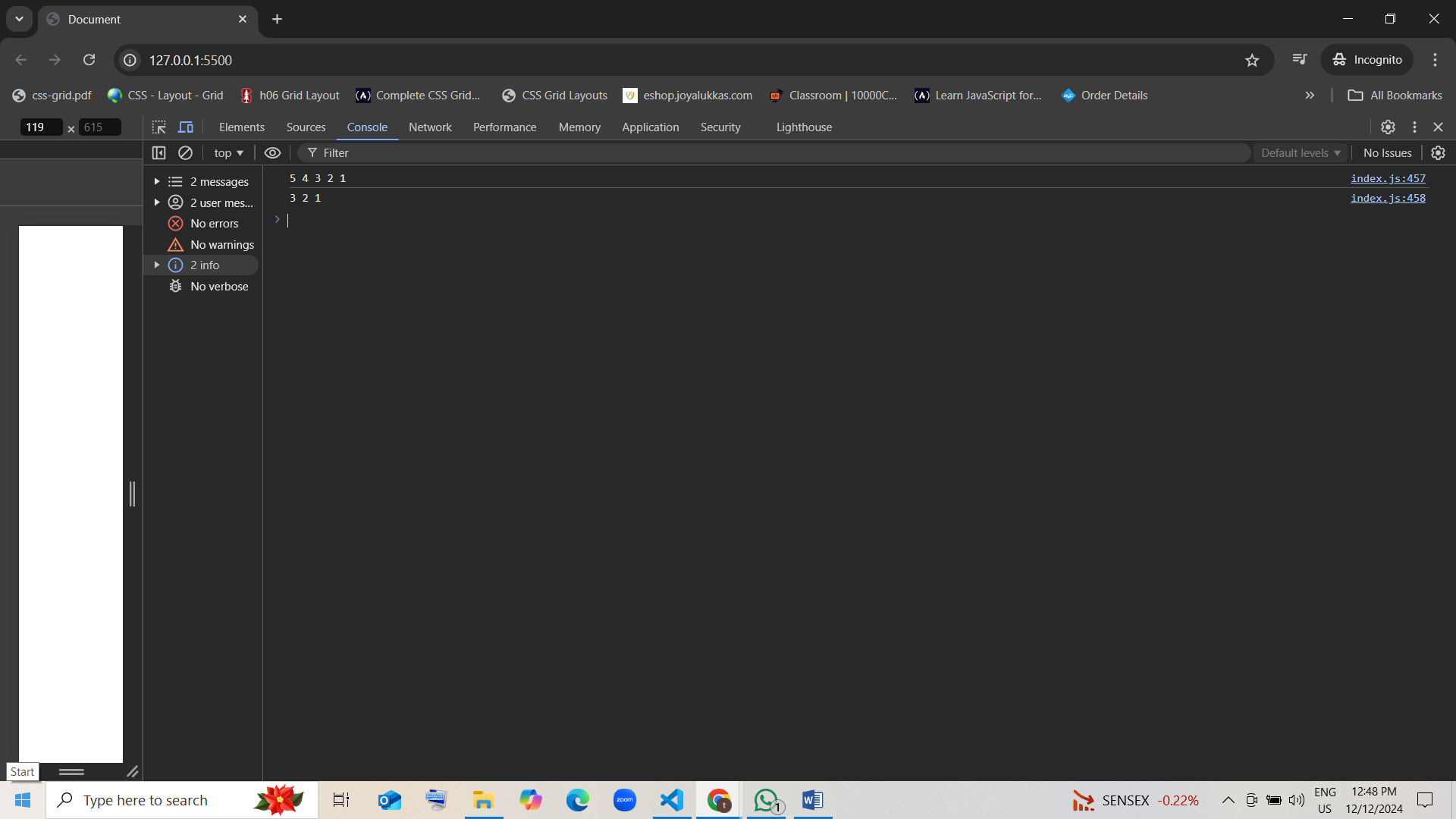
1. **Nested Function with Return**

* function outerFunction(x) {  
   function innerFunction(y) {  
   return x \* y;  
   }  
   return innerFunction;  
  }  
    
  const multiplyBy5 = outerFunction(5);  
  console.log(multiplyBy5(3));  
  console.log(multiplyBy5(4));
* What will be the output of the code above?
  1. 15 and 20 [correct]
  2. 5 and 4
  3. 25 and 20
  4. 15 and 16



1. **Function with While Loop**

* function countDown(n) {  
   let result = "";  
   while (n > 0) {  
   result += n + " ";  
   n--;  
   }  
   return result;  
  }  
    
  console.log(countDown(5));  
  console.log(countDown(3));
* What will be the output of the code above?
  1. "5 4 3 2 1 " and "3 2 1 " [correct]
  2. "5 4 3 2 " and "3 2 "
  3. "5 4 3 " and "3 2 1 "
  4. "5 " and "3 "



1. **Return Type in Functions Using Do-While Loop**

* function printNumbers(n) {  
   let i = 1;  
   let result = "";  
   do {  
   result += i + " ";  
   i++;  
   } while (i <= n);  
   return result;  
  }  
    
  console.log(printNumbers(4));  
  console.log(printNumbers(2));
* What will be the output of the code above?
  1. "1 2 3 4 " and "1 2 " guess [correct]
  2. "4 3 2 1 " and "2 1 "
  3. "1 2 " and "1 2 3 4 "
  4. "4 " and "2 "

