# **Disclaimer:**

COPYRIGHT NOTICE

©All content, materials, and intellectual property contained within this document, including but not limited to text, images, diagrams, graphics, code, and any other original works, are protected by copyright laws and international treaties. This intellectual property is the sole and exclusive property of SRIKANTH TEKUMUDI

Unauthorized use, reproduction, distribution, modification, or transmission of any portion of this document, in any form or by any means, electronic or mechanical, without the prior written permission of SRIKANTH TEKUMUDI, is strictly prohibited and may result in severe civil and criminal penalties.

By accessing, viewing, or using this document, you acknowledge and agree to abide by all applicable copyright laws and the terms and conditions outlined herein. This copyright notice serves as a legal warning and reminder that the unauthorized use of this intellectual property is strictly prohibited and will be vigorously enforced.

Thank you for respecting the intellectual property rights of SRIKANTH TEKUMUDI

# **Try to answer these questions**

In interviews, these questions are commonly posed to assess your foundational knowledge and understanding.

| let person1 = {  name: 'ravi',  age:21 } let person2 = {  name: "ram",  age:43 } person1=Object.freeze(person1) person2 = Object.seal(person2)  person1.age=31 person2.age = 53  person1.location = 'US' person2.location = 'UK'  console.log(person1.location,person1.age) console.log(person2.location,person2.age) |
| --- |

| function bark() {  console.log('Woof!'); }  bark.animal = 'dog'; |
| --- |

| function getAge() {  'use strict';  age = 21;  console.log(age); }  getAge(); |
| --- |

| function sayHi() {  return (() => 0)(); }  console.log(typeof sayHi()); |
| --- |

| const numbers = [1, 2, 3]; numbers[10] = 11; console.log(numbers); |
| --- |

| {  function display() {  var a = (b = 10);  *//b=10;*  *// var a = b;*  }  display();  console.log(typeof b === "undefined");  console.log(typeof a === "undefined"); } |
| --- |

| for (var i = 0; i < 3; i++) {  setTimeout(() => console.log(i), i\*10);  } |
| --- |

| var num = 8; var num = 10;  console.log(num); |
| --- |

| const foo = () => console.log('First'); const bar = () => setTimeout(() => console.log('Second')); const baz = () => console.log('Third');  bar(); foo(); baz(); |
| --- |

| button.addEventListner('click', () => {  Promise.resolve().then(() => console.log('MircroTask 1'))  console.log("Listner 1")  })  button.addEventListner('click', () => {  Promise.resolve().then(() => console.log('MircroTask 2'))  console.log("Listner 2")  })    button.click() |
| --- |

| console.log('one'); setTimeout(function() {  console.log('two'); }, 0); Promise.resolve().then(function() {  console.log('three'); }) console.log('four'); |
| --- |

| var a = 1;  if(true){  function a(){};  var a = 10;  }  console.log(a) |
| --- |

console.log(typeof foo)

| function foo() {  console.log("1");  }   var foo = 9;   console.log(typeof foo);   *//function number* |
| --- |

| var name = "outer";   function foo() {  console.log(name);  var name = "inner";  }  foo();   *//Error The function declaration in the block uses ES6 declaration semantics (like let or const), which does not allow re-declarations.* |
| --- |

| const a = { foo: 123 }; *// Create an object 'a' with a property 'foo' set to 123* const b = Object.create(a); *// Create an object 'b' that inherits from 'a'* b.foo = 444; *// A - Set the 'foo' property of 'b' to 444* delete b.foo; *// B - Delete the 'foo' property from 'b'*   *//answer* const a = { foo: 123 }; const b = Object.create(a);  console.log(b.foo); *// Initially, b.foo is 123 (inherited from a)*  b.foo = 444; console.log(b.foo); *// A - b.foo is now 444 (direct property of b)*  delete b.foo; console.log(b.foo); *// B - b.foo is 123 (inherited from a again after* |
| --- |

Guess output

| var length = 4; function callback() {  console.log(this.length); *// What is logged?* } const object = {  length: 5,  method() {  arguments[0]();  } }; object.method(callback, 1, 2);  //3  var length = 4; function callback() {  console.log(this.length); *// What is logged?* } const object = {  length: 5,  method(callback) {  callback();  } }; object.method(callback, 1, 2);  //4 |
| --- |

Guess output

| function x() {  a()  function a() {console.log('m')}  a()  function a() {console.log('n')}  a() } x();   n n n |
| --- |

Guess output

| async function data(){    console.log("3")    await new Promise((res,rej)=>setTimeout(res,1000))  console.log("4")   } console.log("1") data() console.log("2")   The async function itself starts executing immediately when called. The code inside the async function runs synchronously until the first await. The async function returns a promise immediately upon being called. The function pauses at await and resumes once the awaited promise resolves. |
| --- |

Guess output

| console.log("Try programiz.pro"); console.log("1")  setTimeout(()=>{  console.log("3") })  Promise.resolve("4").then((data)=> setTimeout(()=>{console.log('4') })) Promise.resolve("5").then((data)=>console.log(data))   console.log("2")  output: 1 2 5 3 4  console.log("1") is executed. Output: 1 The setTimeout callback console.log("3") is scheduled in the macro task queue. The promise Promise.resolve("4") is resolved immediately. The .then callback is scheduled as a micro task. console.log("2") is executed. Output: 2 The micro task (the .then callback) runs, scheduling a setTimeout to log 4 in the macro task queue. The event loop then processes the macro task queue: console.log("3") from the first setTimeout. Output: 3 console.log("4") from the second setTimeout. Output: 4 |
| --- |

Guess output

| let prom1=Promise.resolve(2) let prom2=Promise.resolve(4)  let prom3=Promise.reject("232"); let prom4=Promise.resolve('5')  async function getData(){    let data1,data2  *// try{*  data1= await Promise.all([prom1,prom2])    data2= await Promise.all([prom3,prom3]).catch(ex=>{  console.log(ex)  })    *// }*  *// catch(ex){*    *// console.log(ex)*  *// }*    return [data1,data2]    }   (  async ()=>{  let data=await getData()  console.log(data)  }  )() |
| --- |

Guess output

| function Person(firstName, lastName) {  this.firstName = firstName;  this.lastName = lastName; }  const lydia = new Person('Lydia', 'Hallie'); const sarah = Person('Sarah', 'Smith');  console.log(lydia); console.log(sarah);   A: Person {firstName: "Lydia", lastName: "Hallie"} and undefined B: Person {firstName: "Lydia", lastName: "Hallie"} and Person {firstName: "Sarah", lastName: "Smith"} C: Person {firstName: "Lydia", lastName: "Hallie"} and {} D: Person {firstName: "Lydia", lastName: "Hallie"} and ReferenceError |
| --- |

Guess output

| function\* generatorFunction() {  yield 1;  yield 2;  return 3; }  const generator = generatorFunction();  console.log(generator.next()); console.log(generator.next()); console.log(generator.next());  Answer: A: { value: 1, done: false }, { value: 2, done: false }, { value: 3, done: true } |
| --- |