Multi Choice Type Question In this JavaScript code, What will be printed to the console? 1234567 function multiply(a, b) { return a \* b;}var result = multiply(-2, 3.55);console.log(result); Marks : 1 Negative Marks : 0 image Answer here -7.1 "-7.11" NaN undefined

**ChatGPT said:**

Let's solve this step by step.

The code is:

function multiply(a, b) {

return a \* b;

}

var result = multiply(-2, 3.55);

console.log(result);

1. The function multiply takes two arguments and returns their product.
2. The arguments passed are -2 and 3.55.
3. Multiplying them:

−2×3.55=−7.1-2 \times 3.55 = -7.1−2×3.55=−7.1

1. console.log(result) will print -7.1 as a number.

✅ Correct answer: **-7.1**

**You said:**

Multi Choice Type Question In the JavaScript Code, What will be printed to the console? 12345678910111213 function outer() { var x = 'a'; function inner() { console.log(x); } return inner;}var closureFn = outer();closureFn(); Marks : 1 Negative Marks : 0 image Answer here a 'a' undefined ReferenceError: x is not defined

**ChatGPT said:**

Let's analyze the code carefully:

function outer() {

var x = 'a';

function inner() {

console.log(x);

}

return inner;

}

var closureFn = outer();

closureFn();

**Step-by-step reasoning:**

1. outer() is called, which defines x = 'a' and the inner function, then returns inner.
2. closureFn now holds the inner function.
3. When closureFn() is called, it executes inner().
4. inner() has access to x because of **closure**, even though outer() has already finished executing.
5. console.log(x) prints the value of x, which is 'a'.

✅ Correct answer: **'a'**

**You said:**

Multi Choice Type Question In this JavaScript, What will be printed to the console? 12345678910111213 function add(x, y) { if (y !== undefined) { return x + y; } return function(y) { return x + y; };}console.log(add(2, 3));console.log(add(2)(3)); Marks : 1 Negative Marks : 0 image Answer here 5 5 5 undefined 5 6 TypeError: add(...) is not a function Clear

**ChatGPT said:**

Let's analyze this carefully. The code is:

function add(x, y) {

if (y !== undefined) {

return x + y;

}

return function(y) {

return x + y;

};

}

console.log(add(2, 3));

console.log(add(2)(3));

**Step-by-step reasoning:**

1. add(2, 3)
   * y is 3, which is not undefined.
   * So it returns 2 + 3 = 5.
2. add(2)(3)
   * add(2) is called first. Here, y is undefined, so it returns a **function**: function(y){ return 2 + y; }.
   * Then this returned function is called with (3). So it computes 2 + 3 = 5.

✅ Correct output:

5

5

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? 1234567891011121314 class Person { constructor(name) { this.name = name; } sayName() { console.log(this.name); }}var person1 = new Person("Alice");var person2 = new person1.constructor("Bob");person2.sayName(); Marks : 1 Negative Marks : 0 image Answer here Alice Bob ReferenceError: person1 is not defined TypeError: person1.constructor is not a constructor

**ChatGPT said:**

Let's analyze the code carefully:

class Person {

constructor(name) {

this.name = name;

}

sayName() {

console.log(this.name);

}

}

var person1 = new Person("Alice");

var person2 = new person1.constructor("Bob");

person2.sayName();

**Step-by-step reasoning:**

1. person1 = new Person("Alice") creates an instance with name = "Alice".
2. person1.constructor points to the Person class itself.
3. new person1.constructor("Bob") is equivalent to new Person("Bob"). It creates a new instance person2 with name = "Bob".
4. person2.sayName() prints this.name, which is "Bob".

✅ Correct answer: **Bob**

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? 123456789101112 var obj1 = { value: 10, getValue: function() { return this.value; }};var obj2 = Object.create(obj1);obj2.value = 20;console.log(obj2.getValue()); Marks : 1 Negative Marks : 0 image Answer here 10 20 undefined TypeError: obj2.getValue is not a function

**ChatGPT said:**

Let's break it down:

var obj1 = {

value: 10,

getValue: function() {

return this.value;

}

};

var obj2 = Object.create(obj1);

obj2.value = 20;

console.log(obj2.getValue());

**Step-by-step reasoning:**

1. obj1 has a property value = 10 and a method getValue.
2. obj2 = Object.create(obj1) creates a new object whose **prototype** is obj1.
3. obj2.value = 20 sets value directly on obj2.
4. obj2.getValue() looks for getValue on obj2. It doesn't exist there, so it checks the prototype obj1 and finds getValue.
5. Inside getValue, this refers to the **calling object**, which is obj2.
6. Therefore, this.value is obj2.value = 20.

✅ Correct answer: **20**

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? 12345678910111213141516 class Rectangle { constructor(width, height) { this.width = width; this.height = height; } get area() { return this.width \* this.height; }}var rectangle = new Rectangle(4, 5);rectangle.area = 20;console.log(rectangle.area); Marks : 1 Negative Marks : 0 image Answer here 20 9 20 (but with a warning) TypeError:

**ChatGPT said:**

Let's analyze the code carefully:

class Rectangle {

constructor(width, height) {

this.width = width;

this.height = height;

}

get area() {

return this.width \* this.height;

}

}

var rectangle = new Rectangle(4, 5);

rectangle.area = 20;

console.log(rectangle.area);

**Step-by-step reasoning:**

1. Rectangle has a **getter** area, but **no setter**.
2. rectangle.area = 20 attempts to set a value to a **read-only getter**. In strict mode, this would throw a TypeError. In non-strict mode, it fails silently and does **not** change the value.
3. console.log(rectangle.area) calls the getter:

rectangle.width∗rectangle.height=4∗5=20rectangle.width \* rectangle.height = 4 \* 5 = 20rectangle.width∗rectangle.height=4∗5=20

So interestingly, even though rectangle.area = 20 is ignored, the getter **still returns 20** because 4 \* 5 = 20.

✅ Correct answer: **20**

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? 12345678910111213141516171819 class Shape { constructor() { this.name = "Shape"; } static printName() { console.log(this.name); }}class Rectangle extends Shape { constructor() { super(); this.name = "Rectangle"; }}Rectangle.printName(); Marks : 1 Negative Marks : 0 image Answer here Shape Rectangle TypeError: Rectangle.printName is not a function ReferenceError: Rectangle is not defined Clear

**ChatGPT said:**

Let's analyze the code carefully:

class Shape {

constructor() {

this.name = "Shape";

}

static printName() {

console.log(this.name);

}

}

class Rectangle extends Shape {

constructor() {

super();

this.name = "Rectangle";

}

}

Rectangle.printName();

**Step-by-step reasoning:**

1. printName is a **static method** of Shape. Static methods are inherited by subclasses.
2. Rectangle.printName() calls the static method on Rectangle.
3. Inside a static method, this refers to the **class itself**, so this.name is Rectangle.name (the name of the class).
4. The name property of the class (not instance) is "Rectangle" automatically.

✅ Correct answer: **Rectangle**

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? console.log(this.name); 123456789101112131415161718192021222324252627282930313233 class Vehicle { constructor() { this.speed = 0; } accelerate() { this.speed += 10; } static getInfo() { console.log("This is a vehicle."); }}class Car extends Vehicle { constructor() { super(); } accelerate() { super.accelerate(); console.log("Accelerating the car."); } static getInfo() { console.log("This is a car."); }}var car = new Car();car.accelerate();Car.getInfo(); Marks : 1 Negative Marks : 0 image Answer here Accelerating the car. This is a vehicle. This is a car. Accelerating the car. Accelerating the car. This is a car. Clear

**ChatGPT said:**

Let's break it down carefully:

class Vehicle {

constructor() {

this.speed = 0;

}

accelerate() {

this.speed += 10;

}

static getInfo() {

console.log("This is a vehicle.");

}

}

class Car extends Vehicle {

constructor() {

super();

}

accelerate() {

super.accelerate();

console.log("Accelerating the car.");

}

static getInfo() {

console.log("This is a car.");

}

}

var car = new Car();

car.accelerate();

Car.getInfo();

**Step-by-step reasoning:**

1. car.accelerate() calls the **instance method** in Car.
   * super.accelerate() runs Vehicle's accelerate() method, increasing speed from 0 to 10.
   * Then it prints "Accelerating the car.".
2. Car.getInfo() calls the **static method** in Car, which prints "This is a car.".

✅ Correct console output:

Accelerating the car.

This is a car.

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? console.log(this.name); 1234567891011 var x = 5;var y = "5";if (x === y) { console.log("A");} else if (x == y) { console.log("B");} else { console.log("C");} Marks : 1 Negative Marks : 0 image Answer here A B C Nothing (no output)

**ChatGPT said:**

Let's analyze the code carefully:

var x = 5;

var y = "5";

if (x === y) {

console.log("A");

} else if (x == y) {

console.log("B");

} else {

console.log("C");

}

**Step-by-step reasoning:**

1. x === y checks **strict equality** (value and type).
   * x is a number (5) and y is a string ("5").
   * Types are different → **false**, so "A" is not printed.
2. x == y checks **loose equality** (type coercion allowed).
   * "5" is converted to number 5.
   * 5 == 5 → **true**, so "B" is printed.

✅ Correct answer: **B**

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? console.log(this.name); 1234567891011121314151617181920212223242526 class Animal { constructor(name) { this.name = name; } speak() { console.log("The animal makes a sound."); }}class Dog extends Animal { constructor(name) { super(name); } speak() { console.log("The dog barks."); }}var animal = new Animal("Leo");var dog = new Dog("Buddy");animal.speak();dog.speak(); Marks : 1 Negative Marks : 0 image Answer here The animal makes a sound. The dog barks. The animal makes a sound. The animal makes a sound. The dog barks. The dog barks. ReferenceError: animal.speak is not a function

**ChatGPT said:**

Let's analyze the code carefully:

class Animal {

constructor(name) {

this.name = name;

}

speak() {

console.log("The animal makes a sound.");

}

}

class Dog extends Animal {

constructor(name) {

super(name);

}

speak() {

console.log("The dog barks.");

}

}

var animal = new Animal("Leo");

var dog = new Dog("Buddy");

animal.speak();

dog.speak();

**Step-by-step reasoning:**

1. animal.speak() calls the speak method of the Animal class → prints: "The animal makes a sound."
2. dog.speak() calls the overridden speak method of the Dog class → prints: "The dog barks."

✅ Correct console output:

The animal makes a sound.

The dog barks.

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed on the console? console.log(this.name); 12345678910111213141516 function divide(a, b) { if (b === 0) { throw new Error("Cannot divide by zero."); } return a / b;}try { var result = divide(10, 0); console.log(result);} catch (err) { console.log(err.message);}console.log(result); Marks : 1 Negative Marks : 0 image Answer here undefined Cannot divide by zero. Cannot divide by zero. undefined "Cannot divide by zero." and undefined SyntaxError: Unexpected identifier Clear

**ChatGPT said:**

Let's analyze the code carefully:

function divide(a, b) {

if (b === 0) {

throw new Error("Cannot divide by zero.");

}

return a / b;

}

try {

var result = divide(10, 0);

console.log(result);

} catch (err) {

console.log(err.message);

}

console.log(result);

**Step-by-step reasoning:**

1. divide(10, 0) is called → b === 0, so an **Error** is thrown.
2. The catch block catches the error and prints err.message → "Cannot divide by zero.".
3. var result was declared, but the assignment never happened because the function threw an error.
   * In JavaScript, a var declared variable is **hoisted** and initialized with undefined.
4. console.log(result) after the try-catch prints undefined.

✅ Correct console output:

Cannot divide by zero.

undefined

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? console.log(this.name); 1234567 function add(a, b) { return a + b;}var result = add(2, "3");console.log(result); Marks : 1 Negative Marks : 0 image Answer here 5 "5" 23 "23"

**ChatGPT said:**

Let's analyze the code carefully:

function add(a, b) {

return a + b;

}

var result = add(2, "3");

console.log(result);

**Step-by-step reasoning:**

1. a = 2 (number), b = "3" (string).
2. The + operator with a number and a string **performs string concatenation**.
   * 2 + "3" → "23" (string).

✅ Correct answer: **"23"**

**You said:**

Multi Choice Type Question In this JavaScript Code, What will be printed to the console? console.log(this.name); 1234567891011 function calculateSum(numbers) { var sum = 0; for (var i = 0; i < numbers.length; i++) { sum += numbers[i]; } return sum;}var numbers = [1, 2, 3, 4, 5];console.log(calculateSum(numbers)); Marks : 1 Negative Marks : 0 image Answer here 10 15 20 25

**ChatGPT said:**

Let's analyze the code:

function calculateSum(numbers) {

var sum = 0;

for (var i = 0; i < numbers.length; i++) {

sum += numbers[i];

}

return sum;

}

var numbers = [1, 2, 3, 4, 5];

console.log(calculateSum(numbers));

**Step-by-step reasoning:**

1. numbers = [1, 2, 3, 4, 5]
2. Loop adds each element to sum:
   * 0 + 1 = 1
   * 1 + 2 = 3
   * 3 + 3 = 6
   * 6 + 4 = 10
   * 10 + 5 = 15

✅ Correct answer: **15**

**You said:**

Multi Choice Type Question In this JavaScript code, what will be printed to the console? console.log(this.name); 123456789 function printColors(colors) { for (var i = 0; i < colors.length; i++) { console.log("Color at index " + i + ": " + colors[i]); }}var colors = ["red", "green", "blue"];printColors(colors); Marks : 1 Negative Marks : 0 image Answer here Color at index 0: red Color at index 1: green Color at index 2: blue Color at index 0: blue Color at index 1: green Color at index 2: red Color at index 0: green Color at index 1: red Color at index 2: blue Color at index 0: blue Color at index 1: blue Color at index 2: blue

**ChatGPT said:**

Let's analyze the code:

function printColors(colors) {

for (var i = 0; i < colors.length; i++) {

console.log("Color at index " + i + ": " + colors[i]);

}

}

var colors = ["red", "green", "blue"];

printColors(colors);

**Step-by-step reasoning:**

1. colors = ["red", "green", "blue"]
2. Loop iterates from i = 0 to i = 2:
   * i = 0 → prints "Color at index 0: red"
   * i = 1 → prints "Color at index 1: green"
   * i = 2 → prints "Color at index 2: blue"

✅ Correct console output:

Color at index 0: red

Color at index 1: green

Color at index 2: blue

**You said:**

Multi Choice Type Question Given the Typescript code, What will TypeScript do with this code? console.log(this.name); 123456789101112131415 function findMax(numbers) { if (numbers.length === 0) { return null; } var max = numbers[0]; for (var i = 1; i < numbers.length; i++) { if (numbers[i] > max) { max = numbers[i]; } } return max;}var numbers = [17, 42, 8, 23, 56];console.log(findMax(numbers)); Marks : 1 Negative Marks : 0 image Answer here 17 42 56 null

**ChatGPT said:**

Let's analyze the code carefully:

function findMax(numbers) {

if (numbers.length === 0) {

return null;

}

var max = numbers[0];

for (var i = 1; i < numbers.length; i++) {

if (numbers[i] > max) {

max = numbers[i];

}

}

return max;

}

var numbers = [17, 42, 8, 23, 56];

console.log(findMax(numbers));

**Step-by-step reasoning:**

1. numbers = [17, 42, 8, 23, 56]
2. numbers.length !== 0, so we skip returning null.
3. Initialize max = 17. Loop through remaining elements:
   * Compare 42 > 17 → update max = 42
   * Compare 8 > 42 → false
   * Compare 23 > 42 → false
   * Compare 56 > 42 → update max = 56
4. Return max = 56.

✅ Correct output: **56**