**Javascript Objects**

**Q1. What is an object in JavaScript?**

**Answer: An object is a collection of properties, where each property is a key-value pair.**

### Q2. How do you create an object in JavaScript?

**Answer: There are different ways to create an object in JavaScript. The simplest way is to use the object literal notation as follows:**

**let person = {name: 'John', age: 30};**

### Q3. How do you access a property of an object in JavaScript?

**Answer: You can access a property of an object using the dot notation or the square bracket notation. For example:**

**console.log(person.name); // Output: John**

**console.log(person['age']); // Output: 30**

### Q4. What is the difference between dot notation and square bracket notation in accessing object properties?

**Answer: The dot notation is used when you know the property name at the time of coding. The square bracket notation is used when you want to access the property dynamically, for example, when the property name is stored in a variable.**

### Q5. How do you add a new property to an object in JavaScript?

**Answer: You can add a new property to an object using the dot notation or the square bracket notation as follows:**

**person.city = 'New York'; // Using dot notation**

**person['country'] = 'USA'; // Using square bracket notation**

### Q6. How do you remove a property from an object in JavaScript?

**Answer: You can remove a property from an object using the delete keyword as follows:**

**delete person.city;**

### Q7. How do you check if an object has a property in JavaScript?

**Answer: You can check if an object has a property using the hasOwnProperty method as follows:**

**console.log(person.hasOwnProperty('name')); // Output: true**

**console.log(person.hasOwnProperty('salary')); // Output: false**

### Q8. How do you loop through the properties of an object in JavaScript?

**Answer: You can loop through the properties of an object using a for...in loop as follows:**

**for(let property in person) {**

**console.log(`${property}: ${person[property]}`);**

**}**

### Q9. What is a constructor function in JavaScript?

**Answer: A constructor function is a function that is used to create new objects with the same properties and methods.**

### Q10. How do you create an object using a constructor function in JavaScript?

**Answer: You can create an object using a constructor function as follows:**

**function Person(name, age) {**

**this.name = name;**

**this.age = age;**

**}**

**let person = new Person('John', 30);**

### Q11. What is the difference between an object created using object literal notation and an object created using a constructor function?

**Answer: An object created using object literal notation is a single instance, while an object created using a constructor function can create multiple instances.**

### Q12. What is prototypal inheritance in JavaScript?

**Answer: Prototypal inheritance is a way of creating new objects based on an existing object. The new object inherits the properties and methods of the existing object.**

### Q13. How do you create an object using prototypal inheritance in JavaScript?

**Answer: You can create an object using prototypal inheritance as follows:**

**let person = {**

**name: 'John',**

**age: 30,**

**greet: function() {**

**console.log(`Hello, my name is ${this.name} and I am ${this.age} years old`);**

**}**

**};**

**let employee = Object.create(person);**

**employee.jobTitle = 'Software Developer';**

### Q14. What is the difference between an instance property and a prototype property in JavaScript?

**Answer: An instance property is a property that is specific to a particular instance of an object. When you create a new instance of an object, that instance will have its own copy of the instance properties. Any changes made to the instance property of one object do not affect the instance properties of other objects.**

**On the other hand, a prototype property is a property that is shared among all instances of an object. When you create an object, you can add properties to its prototype that will be shared by all instances of the object. If you change the value of a prototype property, that change will be reflected in all instances of the object.**

**For example, consider the following code:**

**function Person(name, age) {**

**this.name = name;**

**this.age = age;**

**}**

**Person.prototype.gender = 'unknown';**

**let john = new Person('John', 30);**

**let jane = new Person('Jane', 25);**

**console.log(john.name); // Output: John**

**console.log(john.gender); // Output: unknown**

**console.log(jane.name); // Output: Jane**

**console.log(jane.gender); // Output: unknown**

**Person.prototype.gender = 'male';**

**console.log(john.gender); // Output: male**

**console.log(jane.gender); // Output: male**

**In this example, name and age are instance properties because they are specific to each instance of the Person object. gender is a prototype property because it is shared by all instances of the Person object. When we change the value of the gender property on the prototype, the change is reflected in all instances of the Person object.**

### Q15. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30,**

**greet: function() {**

**console.log('Hello, my name is ' + this.name + ' and I am ' + this.age + ' years old.');**

**}**

**};**

**person.greet();**

**Output: Hello, my name is John and I am 30 years old.**

### Q16. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30,**

**address: {**

**street: '123 Main St',**

**city: 'Anytown',**

**state: 'CA'**

**}**

**};**

**console.log(person.address.city);**

**Output: Anytown**

### Q17. What is the output of the following code?

**let person1 = {**

**name: 'John',**

**age: 30**

**};**

**let person2 = person1;**

**person2.name = 'Jane';**

### Q18. What is the output of the following code?

**console.log(person1.name);**

**Output: Jane**

**let person = {**

**name: 'John',**

**age: 30,**

**greet: function() {**

**console.log('Hello, my name is ' + this.name + ' and I am ' + this.age + ' years old.');**

**}**

**};**

**let greetFunc = person.greet;**

**greetFunc();**

**Output: Hello, my name is undefined and I am undefined years old.**

### Q19. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**console.log(Object.keys(person));**

**Output: ['name', 'age']**

### Q20. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**delete person.age;**

**console.log(person.age);**

**Output: undefined**

### Q21. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**console.log(person.hasOwnProperty('name'));**

**console.log(person.hasOwnProperty('toString'));**

**Output: true**

**Output: false**

### Q22. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**console.log(person.toString());**

**Output: [object Object]**

### Q23. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**person.email = 'john@example.com';**

**console.log(person.email);**

**Output: john@example.com**

### Q24. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**console.log(Object.values(person));**

**Output: ['John', 30]**

### Q25. What is the output of the following code?

**let person = {**

**name: 'John',**

**age: 30**

**};**

**console.log(person.hasOwnProperty('toString'));**

**console.log(Object.prototype.hasOwnProperty('toString'));**

**Output: false Output: true**