# Assignment Questions 6

💡 **Q.1** What’s Constructor and Its Purpose?

Ans.: A constructor is a special function that creates and initializes an object instance of a class. In JavaScript, a constructor gets called when an object is created using the new keyword. The purpose of a constructor is to create a new object and set values for any existing object properties. In a constructor function ‘this’ does not have a value. It is a substitute for the new object. The value of ‘this’will become the new object when a new object is created.

💡 **Q.2** Explain This Keyword and Its Purpose?

Ans.: In JavaScript, the ‘this’ keyword refers to an object, which object depends on how this is being invoked (used or called). The ‘this’ keyword refers to different objects depending on how it is used:

1. In an object method, this refers to the object.
2. Alone, this refers to the global object.
3. In a function, this refers to the global object.
4. In a function, in strict mode, this is undefined.
5. In an event, this refers to the element that received the event.
6. Methods like call(), apply(), and bind() can refer this to any object.

💡 **Q.3** What’s Call, Apply, Bind Method & Difference Between them

Ans.: The call method is used to execute a function by binding the this value and providing a list of arguments as parameters. It returns the value returned by the function. Similarly, the apply method also binds the this value and executes the function, but it accepts a single array object containing arguments.

The apply method returns the value returned by the function as well. It's important to note that call and apply have the same functionality, with the only difference being that call accepts a list of arguments, while apply accepts an array of arguments.

Note : call() and apply() are identical in functionality, the only difference is that call() accepts a list of arguments; whereas, apply() accepts a single array of argument.

On the other hand, the bind method binds the this value to the function and creates a new function that can be invoked separately. It returns this new function, allowing you to provide arguments as needed. The bind method is useful when you want to create a new function with a specific this value, which can be called later with additional arguments.

💡 **Q.4** Explain OOPS ?

Ans.: OOP is a programming technique that groups data (properties) and methods (actions) together inside a box. It demonstrates the pattern of real-world objects. An object is a data structure containing properties and methods. Consider a student. A student will have characteristics like name, roll number, and class, and he will perform an action, let's say, giving an exam. In object-oriented programming, these characteristics are called data variables. These actions are called data methods. A class defined the blueprint of the object. It demonstrates all characteristics its instance can have and all functions/actions it can perform.

💡 **Q.5** Whats Abstraction and Its Purpose?

Ans.: Abstraction in programming is about simplifying complex systems by concentrating on the most important parts while hiding the intricacies. Its goal is to make code easier to comprehend and maintain. Abstraction achieves this by providing a clear and concise view of objects or systems, separating their inner workings from the external interface. The main purposes of abstraction are to simplify and manage complexity, foster modular design, facilitate code reuse, and enhance the organization and scalability of software systems. In essence, abstraction allows developers to work with a simplified representation of complex systems, enabling better understanding and more efficient development.

💡 **Q.6** Whats Polymorphism and Purpose of it?

Ans.: Polymorphism is a concept used in the object-oriented paradigm that enables us to use the same function in different forms. This reduces repetition and makes the code snippet useful in many different cases. Polymorphism is implemented in JavaScript by generic, overloading, and structural sub-typing. Let us see them all in detail.

💡 **Q.7** Whats Inheritance and Purpose of it?

Ans.: We can create classes in JavaScript using all the functionality from another class, which is then called its parent class. The child class will contain all the data variables and data methods, and we can add more to it. Basically, inheritance empowers us to reuse code efficiently. We can implement inheritance in JavaScript using the extends keyword. JavaScript lets objects inherit properties from parent objects or any other objects. It uses the concept of prototypal inheritance.

💡 **Q.8** Whats Encapsulation and Purpose of it ?

Ans.: Encapsulation puts the data variables and the data functions together inside a box. Encapsulation ensures that data can only be accessed using the data functions defined inside the class, and abstraction ensures not anyone outside this encapsulated box can access it. Encapsulation wraps the data variables and data functions/methods together inside a single box/unit. This pillar of oops ensures that oops classes and object resembles real-world objects. In encapsulation, we group data and actions together so that data can be secured.

💡 **Q.9** Explain Class in JavaScript?

Ans.: JavaScript is a prototype-based language, and it doesn't have classes in it. We define the templates for objects using constructor functions or prototypes. But JavaScript does have support for the class keyword. The keyword class in JavaScript is actually a syntactic sugar over the prototype-based inheritance, which is already present and supported in JavaScript. JS provides it so that the object creation aligns with other object-oriented languages where the creation of classes is supported.

💡 **Q.10** What’s Super Keyword & What it does?

Ans.: The super keyword is a tool used in object-oriented programming to refer to the parent class or superclass from within a derived class. It allows the derived class to access and use the methods and variables of its parent class. This is especially helpful when the derived class wants to modify or extend the behavior of the parent class. By using the super keyword, the derived class can override methods from the parent class while still retaining the original functionality. It can also utilize the constructors of the parent class to ensure proper initialization of inherited members. In essence, the super keyword enhances the flexibility and adaptability of object-oriented programs by facilitating communication and cooperation between related classes.