# **Solutions**

1. **Callback:**

A callback is a function that is passed as an argument to another function that executes the callback based on the result. They are basically functions that are executed only after a result is produced. Callbacks are an important part of asynchronous JavaScript.

**Callback Hell:**

Callback hell is a group of callbacks arranged in one after another, Every callback depends on the previous callbacks, they can only executes when previous callbacks are executed,

1. We can Iterate an object in two ways in Javascript
2. Using for loop
3. By using Entry method

**Using For Loop:**

Using for loop we can iterate an Object in Javascript. In each iteration, you can get the object key and by using that you can access the property value.

**Example**:

var person = {

firstName: “Govardhan”,

lastName: “Reddy”,

age: 22,

company: “SNAD”

};

for (var key in person) {

if (person.hasOwnProperty(key)) {

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

}

}

**Using Entry Method:**

Object.entries() takes input as object it will return array of objects won string-keyed property.

Example:

var person = {

firstName: “Govardhan”,

lastName: “Reddy”,

age: 22,

company: “SNAD”

};

Object.entries(person).forEach(([key, value]) => {

console.log(`${key}: ${value}`);

});

1. **Promises:**

 A **Promise** is a proxy for a value not necessarily known when the promise is created. It allows you to associate handlers with an asynchronous action's eventual success value or failure reason. This lets asynchronous methods return values like synchronous methods: instead of immediately returning the final value, the asynchronous method returns a promise to supply the value at some point in the future.

A Promise is in one of these states:

* **pending**: initial state, neither fulfilled nor rejected.
* **fulfilled**: meaning that the operation was completed successfully.
* **rejected**: meaning that the operation failed.

Exmple:

let promise = new Promise((resolve, reject) => {

            let a = 10;

            let b = 100;

            let c = a / b;

            if (a > b)

                resolve(“a is greater”);

            else

                reject("invalid Data");

        });

        promise

            .then(data => console.log(data))        //fulfilled

            .catch(reason => console.log(reason))   //rejected

            .finally(() => console.log("Finally block")); //always

1. **Implicit Type Conversion:**

In certain situations, JavaScript automatically converts one data type to another (to the right type). This is known as implicit conversion.

Example:

// numeric string used with + gives string type

let result;

result = '3' + 2;

console.log(result) // "32"

result = '3' + true;

console.log(result); // "3true"

result = '3' + undefined;

console.log(result); // "3undefined"

result = '3' + null;

console.log(result); // "3null"

When a number is added to a string, JavaScript converts the number to a string before concatenation.

1. Difference between js and ts

Javascript:

* it is aprototype based language
* it does not support Static typing
* it does not support inter face
* its does not support modularity
* it is loosely typed scripting language

Typescript

* it is object oriented language
* its support static typing
* its support interface
* its support modularity
* it is a strongly type scripting language

7. Currying:

Currying simply means evaluating functions with multiple arguments and decomposing them into a sequence of functions with a single argument.

In other terms, currying is when a function — instead of taking all arguments at one time — takes the first one and returns a new function, which takes the second one and returns a new function, which takes the third one, etc. until all arguments are completed.

1. Pass by Reference:

Pass-by-reference means to pass the reference of an argument in the calling function to the corresponding formal parameter of the called function. The called function can modify the value of the argument by using its reference passed in.

Pass By Value:

pass by value means you are making a copy in memory of the actual parameter's value that is passed in, a copy of the contents of the actual parameter. Use pass by value when when you are only "using" the parameter for some computation, not changing it for the client program.