In order to print an array, no need for a for-loop to print an array.

console.log(array) will print the array.

**Promises**

Promise is used for asynchronous functions. A promise always requires a function, and the function is then called. A promise can have any function. It can also have anonymous function with resolve or rejected as the argument(for example, in file reading).

The function, which is given in the promise argument, always gets executed. This is the callback function. Once it gets executed, it calls resolve(It passes successfully) or rejected(it failed).

If .then() is there in the program, the call stack goes from promise to .then() when the promise gets resolved.

The basic crux is that the function, in the promise argument, on getting executed passes resolve. Once resolved, it goes to the .then() call.

As mentioned earlier, the function can be anything, it can be an anonymous function also.

If an instance of the promise is called, before the function calls ‘resolved’, the output will be pending. And if an instance of the promise is called, after the function calls ‘resolved’, then the output will be resolved.

When you have to pass a function, which you have already created before, in promise then use anonymous function in the argument of promise, and then execute the function inside, and resolve() it.

We can also pass an anonymous function in the .then() function.

For eg:

p.then(function(){

console.log(p);

})

You can also wrap your promise function in a function. Eg:

function temp(){

let p=new Promise(function(resolve){

resolve("Hello World");

})

return p;

}

function callback(val){

console.log(val);

}

temp().then(callback);

**Arrow Function**

In arrow function, we write the arguments in the bracket and then give =>

We do this usually for anonymous functions. For eg:  
app.get(“/route”,(req,res)=>{

})

Arrow Function-

<https://chat.openai.com/share/587303c1-9c3f-4581-a6f7-d02407bee9f8>