**Function**

In JavaScript, a function is a block of code that is designed to perform a specific task.

## **Function Types in JavaScript**

1. Named Functions
2. Anonymous Functions
3. Arrow Functions
4. Immediately Invoked Function Expressions (IIFE)
5. Higher-Order Functions
6. Async Functions
7. Constructor Function

**Named Functions**

In JavaScript named functions are functions that have a specific name assigned to them when they are defined. Naming functions can make your code more readable, help with debugging, and make it easier to track function calls.

function printName(name) {

console.log("Hello " + name );

}

printName (" mahesh ");

**Anonymous Functions**

In JavaScript, anonymous functions are functions that do not have a name. They are often used as arguments to other functions, assigned to variables, or passed as callbacks. Anonymous functions are particularly useful for short, one-off operations where giving the function a name would be unnecessary.

const printName = function(name) {

console.log("Hello, " + name );

};

printName ("mahesh");

**Arrow Functions**

Arrow functions that provide a more concise syntax for writing functions in JavaScript. They simplify the function declaration process and are particularly useful for writing short, one-liner functions.

const add = (a, b) => a + b;

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

**Immediately Invoked Function Expressions (IIFE)**

An Immediately Invoked Function Expression (IIFE) is a function in JavaScript that is defined and executed immediately after its creation. It is a design pattern used to create a local scope in JavaScript, which helps avoid polluting the global namespace and keeps variables contained within the function's scope

(function() {

console.log("Hi I am IIFE!");

})();

**Higher-Order Functions**

These functions are powerful tools that enable you to write more abstract, flexible, and reusable code. Higher-order functions are a key concept in functional programming and are commonly used in JavaScript for handling tasks such as data manipulation and event handling.

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

const squaredNumbers = numbers.map(function(num) {

return num \* num;

});

console.log(squaredNumbers); // Output: [1, 4, 9, 16, 25]

**Constructor Function**

In JavaScript, you can create objects using functions combined with the new keyword. This approach is known as constructor functions. Constructor functions are used to create and initialize new objects and are a fundamental part of object-oriented programming in JavaScript.

function Person(name, age) {

this.name = name;

this.age = age;

this.sayHello = function() {

console.log("Hello, my name is " + this.name);

};

}

**Async Functions**

Async functions in JavaScript are a feature introduced in ES2017 (ES8) that make handling asynchronous operations more straightforward and readable. They simplify the process of working with asynchronous code by allowing you to write code that looks and behaves like synchronous code. This helps avoid callback hell and makes code more manageable

async function sayHello() {

return "Hello";

}

sayHello().then(message => console.log(message));