JavaScript Functions

1. Function Declaration

A function is defined using the function keyword, followed by a name, parentheses ( ), and a block of code enclosed in curly braces { }.

Syntax

function greet(name) {

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

}

greet("Alice");

1. Function Expression

A function expression defines a function within an expression. These are often anonymous functions assigned to variables.

Syntax

const greet = function(name)

{

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

};

greet("Bob");

1. Arrow Functions

Arrow functions provide a more concise syntax for writing functions. They are often used in situations where a function is short, such as in callbacks or simple transformations.

Syntax

const greet = (name) => {

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

};

greet("Charlie");

1. Function Parameters and Arguments

Functions can accept parameters, which act as placeholders for values passed into the function when it is called.

Syntax

function add(a, b) {

return a + b;

}

console.log(add(3, 4));

1. Return Statement

The return keyword is used to specify the value that a function should return to the caller. If no return is specified, the function returns undefined by default.

Syntax

function square(x) {

return x \* x;

}

console.log(square(4));

1. IIFE (Immediately Invoked Function Expressions)

An IIFE is a function that is defined and executed immediately. It is often used to create a local scope to avoid polluting the global scope.

Syntax

(function() {

console.log("This function runs immediately!");

})();

1. Anonymous Functions

An anonymous function is a function without a name. It is commonly used in callbacks and event handlers.

Syntax

setTimeout(function() {

console.log("This is an anonymous function!");

}, 1000);