# What is Function Overloading and Overriding in JavaScript?

## ☑️ Function Overloading

### 🔘 Definition:

**Function Overloading** means having **multiple functions** with the **same name** but **different parameters** (type or count).It's common in languages like **Java or C++**, but **JavaScript does NOT support it natively**.

### ☑️ Example in Other Languages (Java-like):

```
void greet()          // No parameter
void greet(String name) // One parameter
```

But in **JavaScript**, if you define functions with the same name, the **last one will overwrite the previous ones**.

### ❌ Not True Overloading in JS:

```
function greet() {
  console.log("Hello");
}

function greet(name) {
  console.log("Hello, " + name);
}

greet();       // Output: Hello, undefined
greet("Milan"); // Output: Hello, Milan
```

🧠 Only the second function greet(name) is used — the first is ignored.

### ☑️ Simulated Overloading in JS (Using if/else):

```
function greet(name) {
  if (name === undefined) {
    console.log("Hello");
  } else {
    console.log("Hello, " + name);
  }
}

greet();        // Output: Hello
greet("Milan"); // Output: Hello, Milan
```

✔️‼️ This is how we simulate **overloading** in JavaScript using **optional parameters**.

## ☑️ Function Overriding

### 🔘 Definition:

**Function Overriding** means **child class redefines** a method that already exists in the **parent class** with the **same name and parameters**.

🧠 It is a form of **runtime polymorphism** (dynamic).

### ☑️ Example in JavaScript:

```
class Animal {
  speak() {
    console.log("The animal makes a sound.");
  }
}

class Dog extends Animal {
  // Overriding the parent method
  speak() {
    console.log("The dog barks.");
  }
}

const myDog = new Dog();
myDog.speak(); // Output: The dog barks.
```