**Bài Tập 1:** Tìm hiểu và so sánh hai loại hàm "function" và "arrow function" kèm ví dụ minh hoạ.

1.**Function** : Hàm hoặc chức năng, còn gọi là chương trình con, là một phần của mã có thể được tái sử dụng bên ngoài hoặc bên trong nó. Hàm này bao gồm một tập hợp các câu lệnh được gọi là cơ thể hàm (function body). Bạn có thể truyền giá trị cho hàm, và hàm cũng có thể trả về một giá trị.

* Cấu trúc của một hàm :

Function *tên hàm (danh sách tham số truyền vào){*

*Phần thân hàm*

*}*

Ví dụ:

function Hello () {

console.log("Hello World");

}

* Các rang buộc về tên hàm :
  + Tên hàm phải được bắt đầu bằng chữ cái (a-z,A-Z)hoặc ký tự \_.Không được bắt đầu bằng số và các ký tự khác \_
  + Ví dụ :
    - \_helloVietnam (đúng)
    - 123123Vietnam(sai)
* Các loại hàm :
  + Hàm có không có tham số truyền vào

function nothing() {

// body

}

* + Hàm có tham số truyền vào

function name(param\_1, ..., pram\_n) {

//code

}

***Trong đó:***

* ***funName****là tên của hàm các bạn muốn đặt.*
* ***param\_1,...,pram\_n****là các tham số mà các bạn muốn truyền vào hàm(không giới hạn số lượng).*
  + Hàm có trả về giá trị và không trả về giá trị
    - Đối với hàm có giá trị trả về thì phải sử dụng từ khóa return
    - Và ngược lại đối với hàm không có giá trị trả về thì không có từ khóa return
    - Ví dụ:

function tinhTong(a, b) {

return a + b;

}

**2.Arrow Function :**

* Arrow function là một cách ngắn gọn để khai báo hàm trong JavaScript, bằng cách sử dụng kí tự "=>" (mũi tên). Cú pháp cơ bản của arrow function là:

let func = (arg1, arg2,..., argN) => expression;

* Arrow function nhận vào một danh sách các tham số (arg1, arg2,..., argN) và thực hiện các công việc trong biểu thức expression, sau đó trả về giá trị của biểu thức đó.
* Trong trường hợp sử dụng dấu ngoặc {} thì cần phải sử dụng từ khóa return để trả về kết quả

let func = (arg1, arg2,..., argN) => {

// Các câu lệnh và logic ở đây

return something;

}

* Arrow function cũng có thể sử dụng tương tự như function expression trong các tình huống khác nhau

**3. So sánh.**

3.1 Cú pháp và Khai báo:

* Function: Để định nghĩa một hàm thông thường, sử dụng từ khóa "function" sau đó là tên hàm và danh sách tham số trong dấu ngoặc đơn, sau đó là một cặp dấu ngoặc nhọn "{}" chứa cơ thể của hàm.
* Arrow Function: Để định nghĩa một arrow function, sử dụng một biểu thức ngắn gọn chứa tham số và biểu thức trả về, bao gồm dấu mũi tên "=>" giữa tham số và biểu thức.

Ví dụ:

// Function thông thường

function regularFunction(x) {

return x \* 2;

}

// Arrow function

const arrowFunction = (x) => x \* 2;

3.2 Phạm vi "this":

* Function: Hàm thông thường có phạm vi riêng, và giá trị của từ khóa "this" bên trong hàm thay đổi dựa trên cách hàm được gọi. Thường được sử dụng để tham chiếu đến đối tượng gọi hàm.
* Arrow Function: Arrow function không có phạm vi riêng, nó sử dụng phạm vi của hàm bên ngoài (lexical scoping), do đó, giá trị của từ khóa "this" bên trong arrow function giữ nguyên giá trị của "this" của hàm gọi nó. Điều này có nghĩa rằng "this" trong arrow function thường trỏ đến đối tượng bên ngoài hàm arrow.

Ví dụ:

const obj = {

name: 'Alice',

regularFunction: function () {

console.log(this.name); // In ra: "Alice"

},

arrowFunction: () => {

console.log(this.name); // In ra: undefined

},

};

obj.regularFunction();

obj.arrowFunction();

const obj = {

name: 'Alice',

regularFunction: function () {

console.log(this.name); // In ra: "Alice"

},

arrowFunction: () => {

console.log(this.name); // In ra: undefined

},

};

obj.regularFunction();

obj.arrowFunction();

3.3 Tham số và "arguments":

* Function: Hàm thông thường có thể chấp nhận một danh sách tham số tùy ý và có thể truy cập tất cả các tham số này bằng từ khóa "arguments". "arguments" là một đối tượng đặc biệt chứa tất cả các tham số truyền vào hàm.
* Arrow Function: Arrow function chỉ hỗ trợ tham số đơn giản và không hỗ trợ từ khóa "arguments". Arrow function cũng không hỗ trợ rest parameters.

Ví dụ:

// Function với số lượng tham số tùy ý và sử dụng "arguments"

function sum() {

let result = 0;

for (let i = 0; i < arguments.length; i++) {

result += arguments[i];

}

return result;

}

// Arrow function không hỗ trợ "arguments" và chỉ chấp nhận tham số đơn giản

const add = (x, y) => x + y;

3.4 Constructor:

* Function: Hàm thông thường có thể được sử dụng như một constructor để tạo đối tượng mới bằng từ khóa "new". Khi gọi với "new", nó tạo một đối tượng và tham chiếu "this" đến đối tượng đó.
* Arrow Function: Arrow function không thể được sử dụng như một constructor để tạo đối tượng mới.

Ví dụ:

function Person(name) {

this.name = name;

}

const alice = new Person('Alice'); // Tạo đối tượng từ hàm thông thường

const ArrowPerson = (name) => {

this.name = name;

};

// ArrowPerson không thể sử dụng như constructor, sẽ gây lỗi