Iterable và Iterator protocol

- Iterable và Iterator protocol nằm trong nhóm các giao thức **Iteration Protocols**
- Ví dụ một Object được gọi là `Iterable` (Có thể lặp): Array, String, ...
- Object này chỉ cần tuân thủ một số quy tắc sau đây
  - Có hiện thực (khai báo) một phương thức đặc biệt (Method)
  - Method này có `key` là một hằng số `Symbol.iterator` được cung cấp sẵn bởi Javascript.
  - Method này không có tham số truyền vào
  - Method này khi được gọi sẽ `return` về một `Iterator Object`

```
// Cách 1: Khai báo phương thức điHát của đối tượng ChiPu
const chiPu = {
  diHat: function() { /*...*/ }
}

// Cách 2:
const chiPu = {}
chiPu.diHat = function() { /*...*/ }

// Cách 3.1
const chiPu = {}
chiPu['diHat'] = function() { /*...*/ }

// Cách 3.2
const chiPu = {}
const key = 'diHat';
chiPu[key] = function() { /*...*/ }
```

```
// Khai báo Method quy định lại hành vi lặp của Array
const arr = [1,2,3];
arr[Symbol.iterator] = function() {

}
```

- Một Object được coi là một `Iterator` nếu nó tuân thủ các quy tắc sau đây:
  - Có hiện thực hàm `next()` là method của nó
  - Hàm `next()` không nhận vào tham số nhưng khi được gọi sẽ trả về một Object có hai thuộc tính `value` và `done`