Map Function :

- map() is an ECMAScript5 (ES5) feature.
- **map()** method **creates a new array** populated with the results of calling a provided function on every element in the calling array
- map() method is an iterative method.
- map() creates a new array from calling a function for every array element
- map() calls a function once for each element in an array
- map() does not execute the function for empty elements.
- map() does not change the original array.
- Since map builds a new array, calling it without using the returned array is an anti-pattern; use forEach instead.

Example 1 :

```
const array1 = [1, 4, 9, 16];
const map1 = array1.map(x => x * 2);
console.log(map1); // Expected output: Array [2, 8, 18, 32]
```

Syntax :

`map((element) => { /* … */ })` element is The current element being processed in the array.
`map((element, index) => { /* … */ })` index is Optional. The index of the current element.
`map((element, index, array) => { /* … */ })` The array map() was called upon.
`map(callbackFn)`
`map(callbackFn, thisArg)` thisArg is Optional A value to use as this when executing callbackFn.
`map(function (element) { /* … */ })`
`map(function (element, index) { /* … */ })`

Example 2 :

```
const persons = [
{firstname : "Malcom", lastname: "Reynolds"},
{firstname : "Kaylee", lastname: "Frye"},
{firstname : "Jayne", lastname: "Cobb"}
 ];

 persons.map(getFullName);

 function getFullName(item) {
    return [item.firstname,item.lastname].join(" "); // Expected output String : Malcom Reynolds,Kaylee Frye,Jayne Cobb
 }
```

forEach :

- The forEach() method calls a function for each element in an array.
- The forEach() method is not executed for empty elements.
- The forEach() method executes a provided function once for each array element.

Example 1 :

```
let sum = 0;
const numbers = [65, 44, 12, 4];
numbers.forEach(myFunction);    //output : 125 (65+ 44+ 12+ 4)

function myFunction(item) {
  sum += item;
}
```

Syntax : (same as map)

`foEach((element) => { /* … */ })` element is The current element being processed in the array.
`foEach((element, index) => { /* … */ })` index is Optional. The index of the current element.
`foEach((element, index, array) => { /* … */ })` The array foEach() was called upon.
`foEach(callbackFn)`
`foEach(callbackFn, thisArg)` thisArg is Optional A value to use as this when executing callbackFn.
`foEach(function (element) { /* … */ })`
`foEach(function (element, index) { /* … */ })`

foreach vs map

forEach :

- Returns undefined
- executes a provided function once per array element

map :

- creates a new array with the results of calling a provided function on every element in this array
- Returns new array with transformed elements, leaving back original array unchanged.