Arrays

Lesson Objectives

- Learn to use array in JavaScript programming.
- Learn useful array methods

Definition

- Array is a fundamental data structure that can hold multiple elements in a consecutive memory location.
- Memory location of an array is indexed, 0 being the first index for historical reason.

- In JavaScript, arrays are dynamic in both length and types of elements it can hold.

Declaring an Array

- Using array literal syntax

```
const numbers = [];
const fruits = ["Apple", "Banana", "Mango"];
```

- Array being an object type can also be created using new keyword

```
const numbers = new Array(6);
```

Pencil Exercise

- Lesson 10 - Example 6

Array Length

- In JavaScript, arrays have built-in property, length; which represents the current size of the array.

```
let numbers = []
console.log(numbers.length); // 0
numbers = [1,2,3];
console.log(numbers.length) // 3
```

Using an Array

- Array elements can be both accessed and modified using the array index.

```
const students = ["Jim", "Jack", "Jill"];
const student = students[1];
console.log(student);
students[0] = "John";
console.log(students[0])
```

Pencil Exercise

- Lesson 10 – Example 9

Filling an Array

- Loops can be used to fill an array with some default values, usually for testing purposes.

```
const scores = [];
for (let i=0; i<10; i++){
        scores[i] = Math.ceil(Math.random()*100);
}

console.log(scores);
```

Exercise

- Write a program to create an array named scores and fill it with 5 test scores 10, 20, 30, 40 and 50.
- Now write a function named findAverage, that takes an array as an argument and return average of the array values.
- Call findAverage function passing array you created in step1 and save the return result in a variable, average.
- Print the average, it should be 30 for this example.
- Create a second array filled with 10 random values between 0 to 10 and find the average of the array values.
- Make sure your program computes correct average for an array of any size.

Main Point

- Using array we can hold number of elements under a single identifier, that eliminates the need of unique identifier for each values. *Science of consciousness, during transcendence we forget our individual identity and be one with the cosmic identity*.

Array methods

- JavaScript provides several useful methods that one can use to manipulate contents of arrays. (Refer chapter 10)

```
const words = ["Cat", "Ball", "Apple", "Dog"];
console.log(words);

const reversed_words = words.reverse();
console.log(reversed_words);

const phrase = words.join(",");
console.log(phrase);

const sorted_words = words.sort();
console.log(sorted_words);
```

Sorting Numbers

- By default, the built-in `sort()` function in JavaScript sorts arrays lexicographically, even when array elements are of number type.
  - We can always pass a comparator function inside `sort()` method, to sort the array differently.

```
const num_array = [1,11,3,14,6,23,9,5];

const default_sorted_array = num_array.sort();
console.log(default_sorted_array); // [ 1, 11, 14, 23, 3, 5, 6, 9 ]

const properly_sorted_array = num_array.sort(numberComparator);
console.log(properly_sorted_array);

function numberComparator(a,b){
        return a-b;
}
```

Exercise

- Refactor number sorting example in prior slide to use anonymous function and then to use arrow function.

map()

- The map() method **creates a new array** with the result of calling a provided function on every element in the calling array.

```
const arr1 = [1,5,7,9];

function doubleEveryElement(arr){
    const temp_arr = [];
    for(let i =0; i<arr.length; i++){
        temp_arr[i] = arr[i]*2;
    }
    return temp_arr;
}

const doubled_arr = doubleEveryElement(arr1);
console.log(doubled_arr);
```

```
const arr1 = [1,5,7,9];

const doubled_arr = arr1.map(n => n*2);
console.log(doubled_arr);
```

Exercise

- Write a program to map ["apple", "ball", "cat"] into a new array that contains number of characters in each word, in our case mapped array would be [5,4,3]
  - First write a function that uses loop for mapping.
  - Then use the `map()` method.
- Hint: like array string also have `length` property that returns length of a string.

filter()

- The filter() method creates new array with all the elements that pass the test implemented by the provided function.

```
const arr = [1,4,2,3,7,8,8,9,12,3];
const odd_arr = arr.filter(n=> n%2 !== 0);
console.log(odd_arr);
```

- Solve it using loop.

Exercise

- Write a program to filter out all negative numbers from a given array of numbers.

Input	Output
[1,2,-1,3,-2,5,6]	[1,2,3,5,6]

reduce()

- The reduce() method executes a reducer function (that you provide) on each element array, resulting a single output value.

```
const arr1 = [1,2,3,4,5,6,7,8,9,10];
const reducer = (accumulator, currentValue) => accumulator + currentValue;
const arr_sum = arr1.reduce(reducer);
console.log(arr_sum);
```

- Solve it using loop.

Exercise

- Reduce array [1,2,3,4,5] into product of all the elements in the array.
- Solve it using loop

for/of Loop

- Loops through the values of an iterable objects like String, Array etc.

```
const myStr = "Hello world";

for(let s of myStr){
  console.log(s);
}

const arr1 = [1,2,3,4];

for(let n of arr1){
  console.log(n*3);
}
```

- Refactor to use normal for loop.

forEach()

- The forEach() method executes a provided function once for each array element.

```
const arr1 = [1,2,3,4];
arr1.forEach(function(element){
  console.log(element);
});
```

- Refactor to use arrow function.

Main Point

- Array in JavaScript provide helper methods that makes array programming a whole lot easier. Make use of these helper methods to accomplish more by doing less. *Science of consciousness, when we are in harmony with the natural laws, our actions require less effort and hence we can achieve more by doing less*.

Assignment

- Reading Chapter 10
- Chapter 10, programming assignments (2 to 7, 10)