Data structures an Algorithms

What is a Data structure ?

It is a way to organize data that enables it to be processed in an efficient time.

What is an algorithm?

An algorithm is a set of instructions for solving a problem or accomplishing a task. One common example of an algorithm is a recipe, which consists of specific instructions for preparing a dish or meal.

Primitive Data Structure

Primitive data structure is the data structure that allows you to store only single data type values.

- string
- number
- bigint
- boolean
- undefined
- symbol
- null

Data structures an Algorithms

Dynamic and weak typing

JavaScript is a dynamic language with dynamic types. Variables in JavaScript are not directly associated with any particular value type, and any variable can be assigned (and re-assigned) values of all types:

let foo = 42	// foo is now a number
foo = "bar"	// foo is now a string
foo = true	// foo is now a boolean

JavaScript is also a weakly typed language, which means it allows implicit type conversion when an operation involves mismatched types, instead of throwing type errors.

Data structures an Algorithms

Non-Primitive Data Structure

Non-Primitive Data structures are data structure which is created using Primitive data structures. There are two types of Non-Primitive Data Structures:-

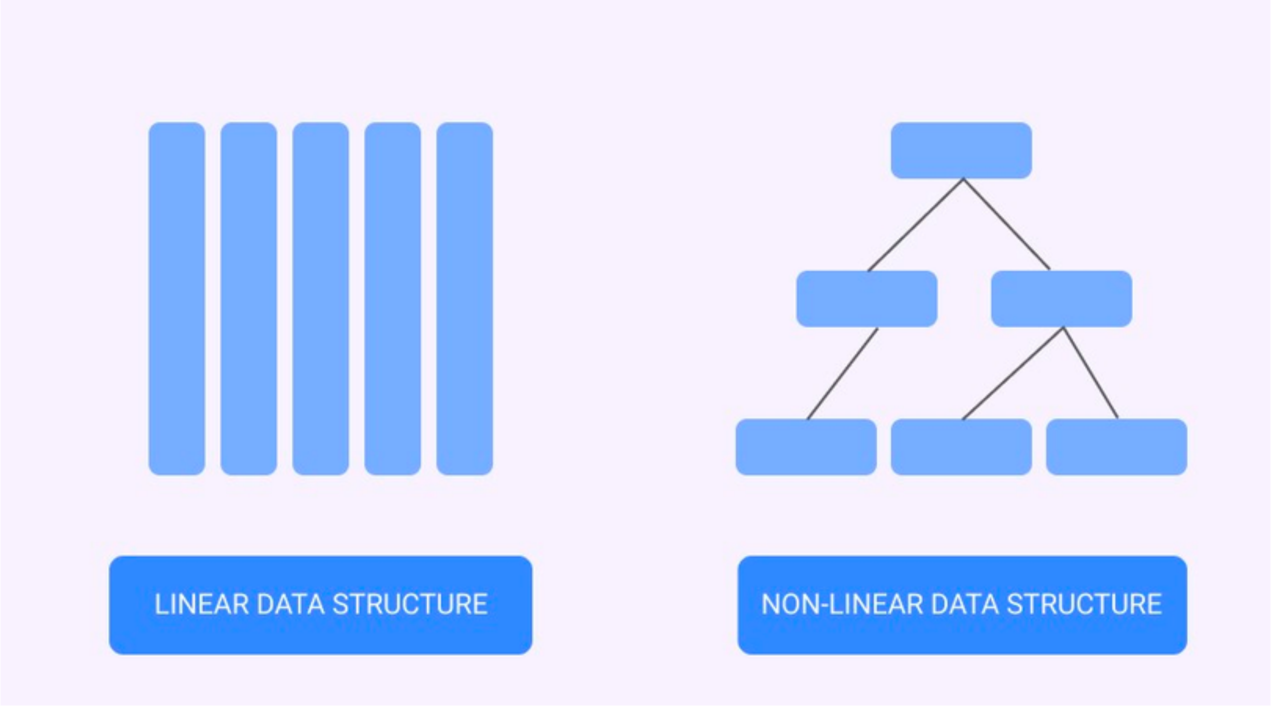
Linear data structure

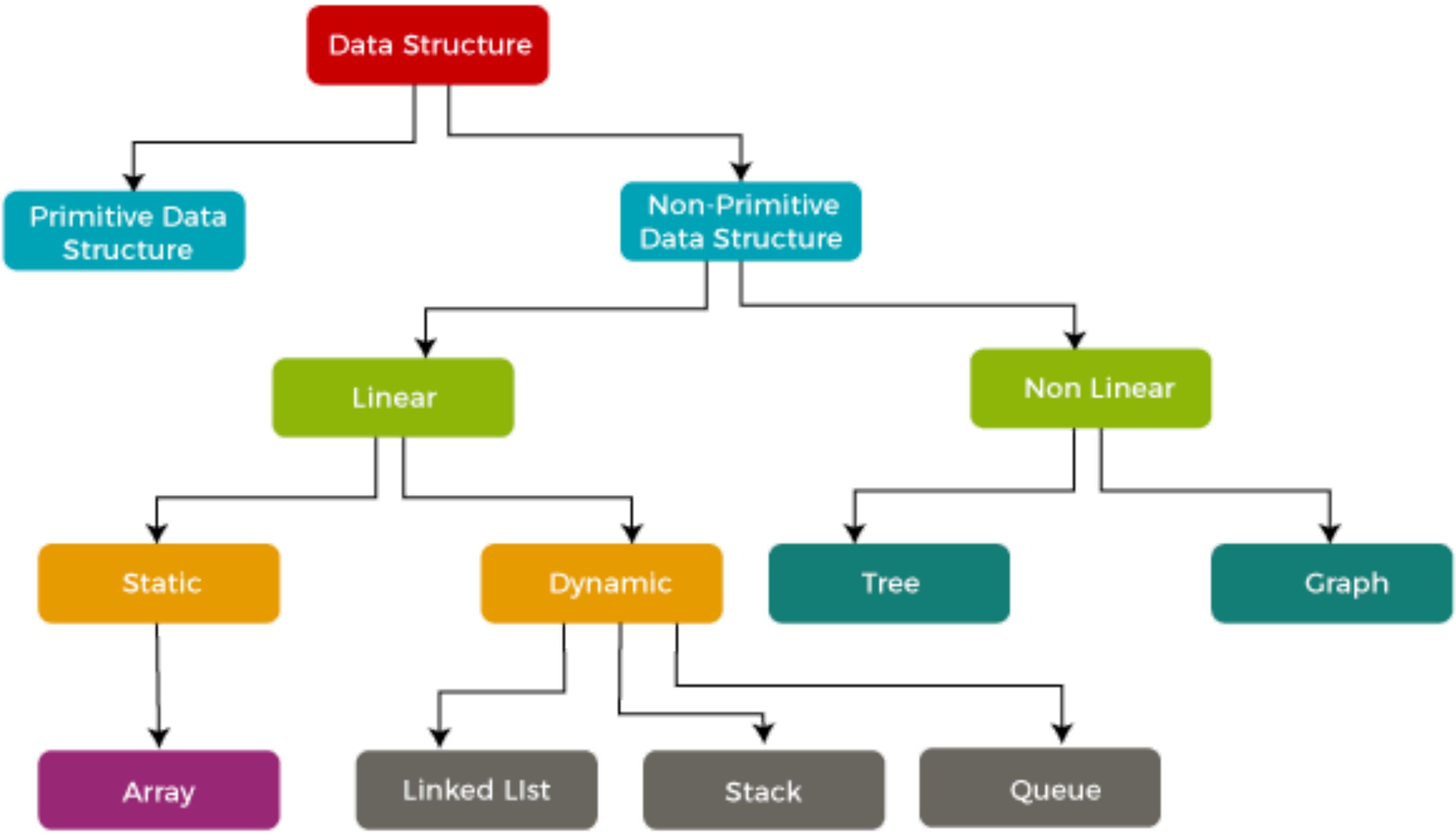
Linear Data structure is a method of organizing data in a linear sequential form.
Like Stack, Queue, Linked list, and Arrays. In these data structures each element is connect
in a sequential manner so that it makes it easier to arrange and implement because the computer's
memory is also arranged in linear form and traverses all elements in a single run.

Non-Linear data structure

Non-Linear Data structure is the data structures that are not organized in linear forms,
like Graphs and Trees. Non-Linear Data structures cannot be easily implemented and It is not
possible to traverse all the elements in a single run, But in Non-Linear Data Structure memory
is utilized in a more efficient way than Linear data structures.

Data structures an Algorithms



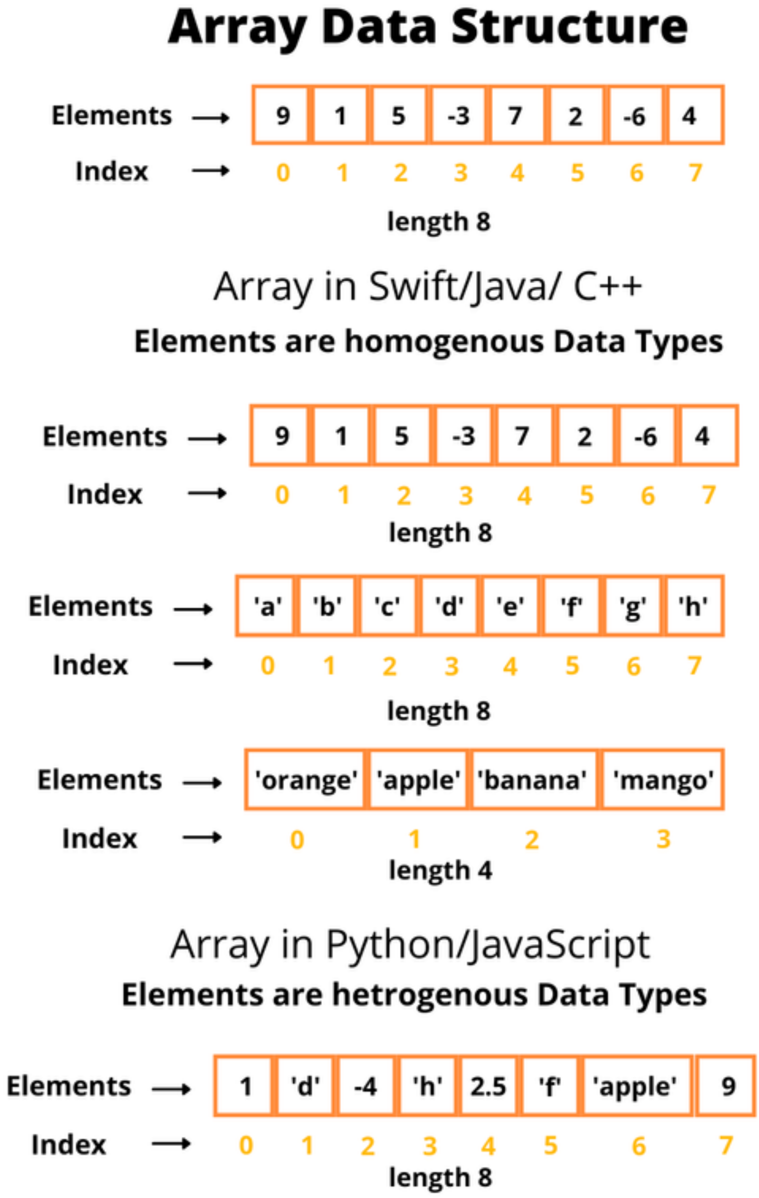


Data structures an Algorithms

ADT(An Abstract Data Type)

Abstract Data type (ADT) is a type (or class) for objects whose behavior is defined by a set of values and a set of operations. The definition of ADT only mentions what operations are to be performed but not how these operations will be implemented. It does not specify how data will be organized in memory and what algorithms will be used for implementing the operations.

It is called "abstract" because it gives an implementation-independent view. So, in order to simplify the process of solving problems, we can create data structures along with their operations, and such data structures that are not in-built are known as Abstract Data Type (ADT).



Array Data Structure

Elements →	9	1	5	-3	7	2	-6	4
Index →	0	1	2	3	4	5	6	7

length 8

Array in Swift/Java/ C++

Elements are homogenous Data Types

Elements →	9	1	5	-3	7	2	-6	4
Index →	0	1	2	3	4	5	6	7

length 8

Elements →	'a'	'b'	'c'	'd'	'e'	'f'	'g'	'h'
Index →	0	1	2	3	4	5	6	7

length 8

Elements →	'orange'	'apple'	'banana'	'mango'
Index →	0	1	2	3

length 4

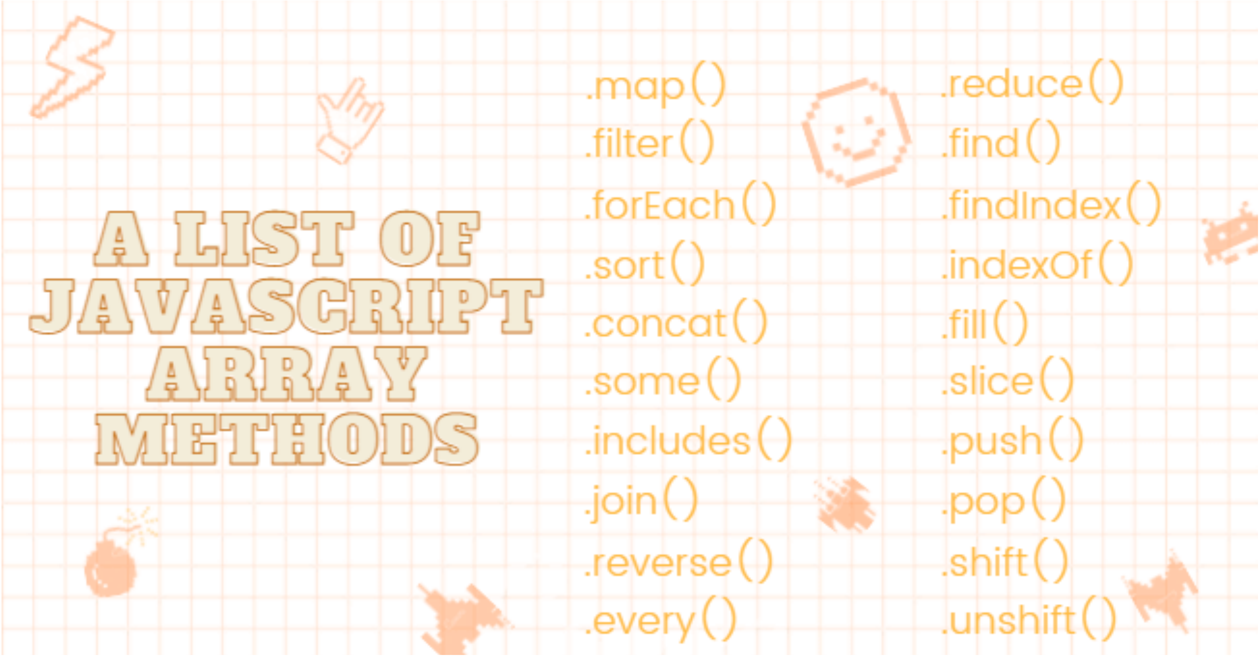
Array in Python/JavaScript

Elements are hetrogenous Data Types

Elements →	1	'd'	-4	'h'	2.5	'f'	'apple'	9
Index →	0	1	2	3	4	5	6	7

length 8

Data structures an Algorithms



array.sort()

sort() method is used to arrange the elements of the given array either in ascending or descending order, by default it arranges the elements in the ascending order.

```
let arr = [4, 2, 5, 1, 3]
let result = arr.sort()

console.log('Orginal array',arr) // "Orginal array", [1, 2, 3, 4, 5]
console.log('New return array',result) // "New return array", [1, 2, 3, 4, 5]

// you can also provide function expressions to determine the order
let arr1 = [6, -3, -10, 0, 2, 8]

// to sort in ascending order
arr1.sort((a,b) => a - b)

console.log(arr1) // [-10, -3, 0, 2, 6, 8]

// to sort in descending order
arr1.sort((a,b) => b - a)

console.log(arr1) // [8, 6, 2, 0, -3, -10]
```

Data structures an Algorithms

array.concat()

concat() method is used to merge two or more arrays and return a new array. It does not affect the original arrays and returns the new array of the sum of all the array lengths.

```
let arr = [1,2,3,4,5]
let arr1 = [6,7,8,9,10]
let arr2 = [11,12,13,14,15]

// merging two arrays
let result1 = arr.concat(arr1)
console.log(arr) // [1, 2, 3, 4, 5]
console.log(arr1) // [6, 7, 8, 9, 10]
console.log(result1) // [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]


//merging two and more arrays
let result2 = arr.concat(arr1,arr2)
console.log(arr) // [1, 2, 3, 4, 5]
console.log(arr1) // [6, 7, 8, 9, 10]
console.log(arr2) // [11, 12, 13, 14, 15]
console.log(result2) // [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

array.some()

concat() method is used to merge two or more arrays and return a new array. It does not affect the original arrays and returns the new array of the sum of all the array lengths.

```
let arr = [2,5,7,4,6,1]

let result = arr.some(item => item > 8)
let result2 = arr.some(item => item % 3 == 0)

console.log(arr) // [2, 5, 7, 4, 6, 1]
console.log(result) // false
console.log(result2) // true
```

Data structures an Algorithms

array.includes()

includes() method is used to determine that a certain value is present among the elements in the given array or not and returns true or false accordingly. It also does not affect the original array.

```
let arr = [2,5,7,4,6,1]

let result = arr.includes(4)
let result2 = arr.includes(9)

console.log(arr) // [2, 5, 7, 4, 6, 1]
console.log(result) // true
console.log(result2) // false
```

array.join()

join() method is used to return a new string, after concatenating all the elements of the given array separated with a specified separator and It does not affect the original array.

```
let arr = [’A’,’p’,’p’,’l’,’e’]
let result = arr.join() // giving no seprator
let result1 = arr.join(”) // giving empty seprator

console.log(arr) // [”A”, ”p”, ”p”, ”l”, ”e”]
console.log(result) // ”A,p,p,l,e”
console.log(result1) // ”Apple”

let arr1 = [’This’,’is’,’so’,’awesome’]

let result2 = arr1.join(’ ’) // giving space as a seprator
let result3 = arr1.join(’-’) // giving - as a seprator

console.log(arr1) // [”This”, ”is”, ”so”, ”awesome”]
console.log(result2) // ”This is so awesome”
console.log(result3) // ”This-is-so-awesome”
```

KEEP
LEARNING
AND
HAPPY
CODING