Google | طويق

Arrays in JavaScript

أكاديمية طويق TUWAIQ ACADEMY

وزارة الاتصالات وتقنية المعلومات MINISTRY OF COMMUNICATIONS AND INFORMATION TECHNOLOGY

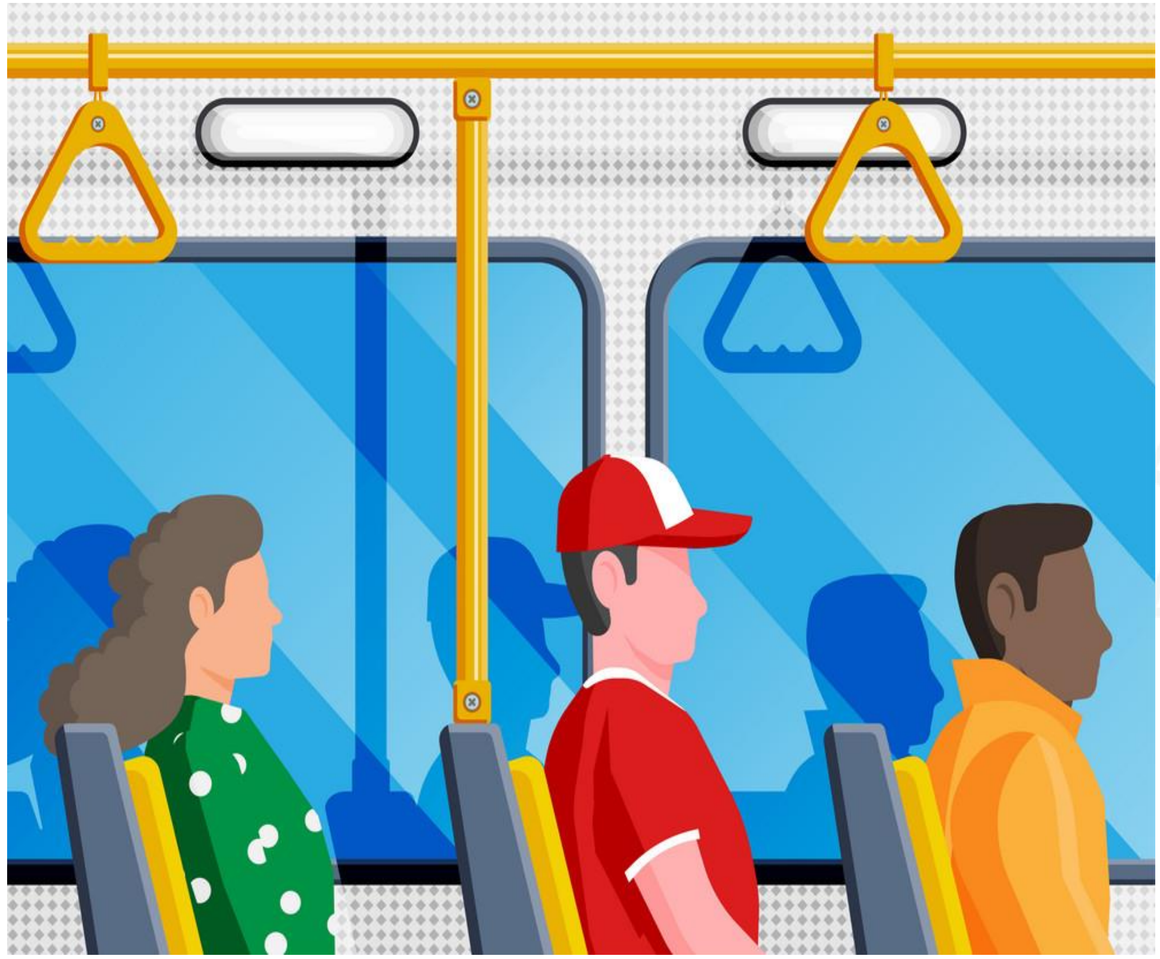
الاتحاد السعودي للأمن السيبراني والبرمجة والدرونز SAUDI FEDERATION FOR CYBERSECURITY, PROGRAMMING & DRONES

مهارات المستقبل | stc

Objective

- Introduction arrays.
- Manipulating arrays.
- Array destructuring
- Sets

What is an array?



- An array is an object which contains a **list of values**, each value is called an element, and each element has a numeric position in the array, known as its **index**.

```
let studentTickets = [10, 20, 15];
```

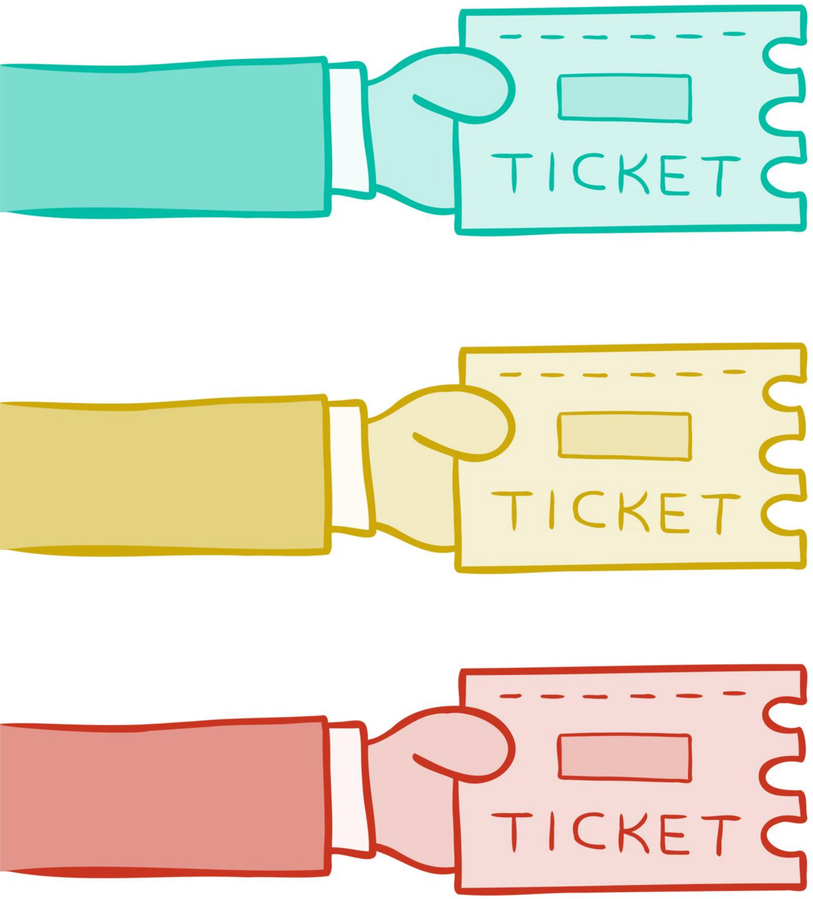
- starting at index **0** and continuing to an index **array.length – 1.**

- Can **access a specific value** stored in an array by using its position in the array or **index**:

```
studentTickets[0];
```

- An element inside an array can be of **any type**, and **different elements** of the **same array can be of different types**.

Manipulating arrays

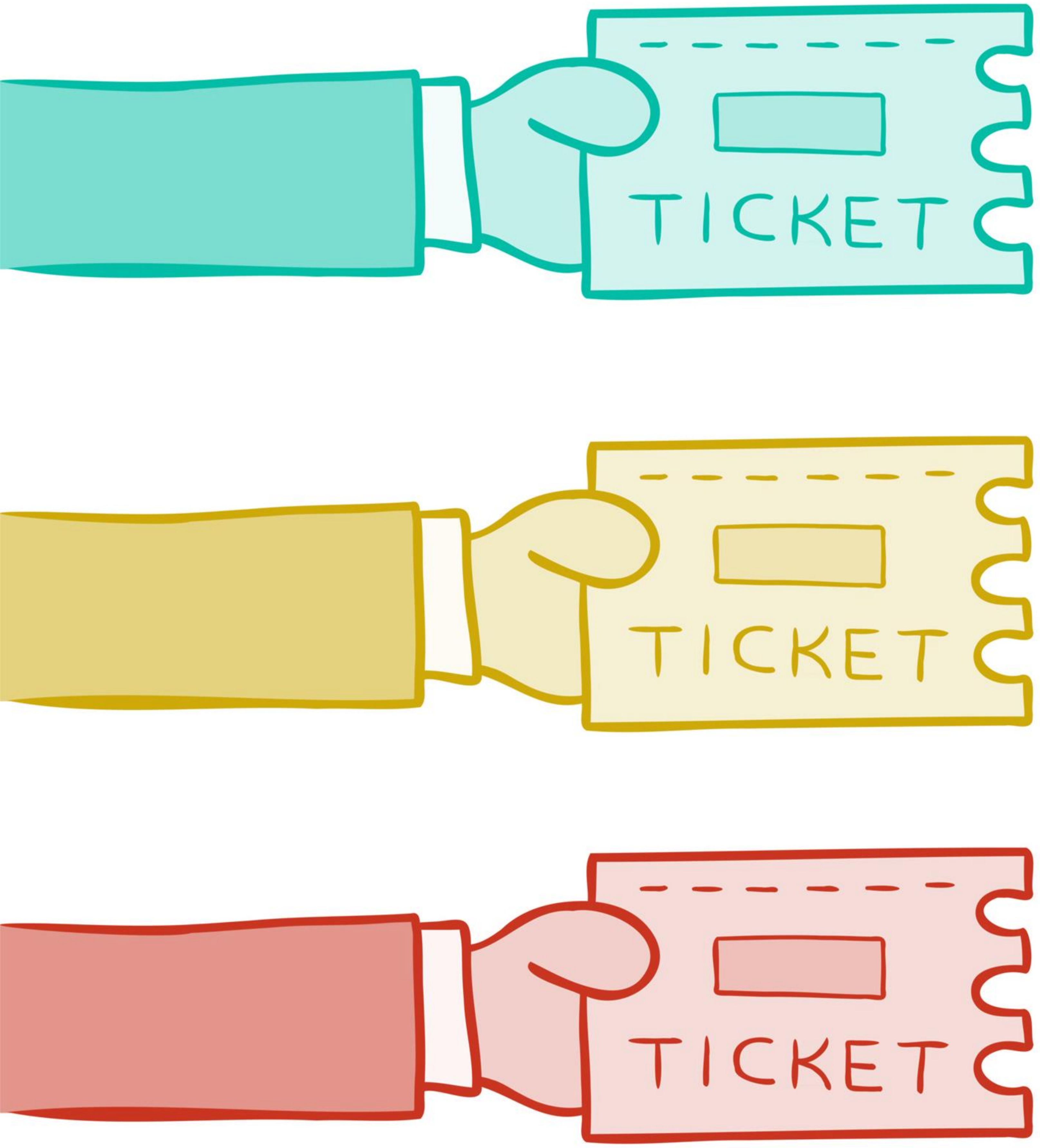


- **for loop:**

```
let studentTickets = [10, 20, 15];

let ticketsPrice =0;

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

 ticketsPrice += studentTickets[i]

 console.log(i + " " +ticketsPrice)

}

console.log(ticketsPrice)
```

https://codepen.io/ecacosca/pen/KKqbQwP?editors=0011

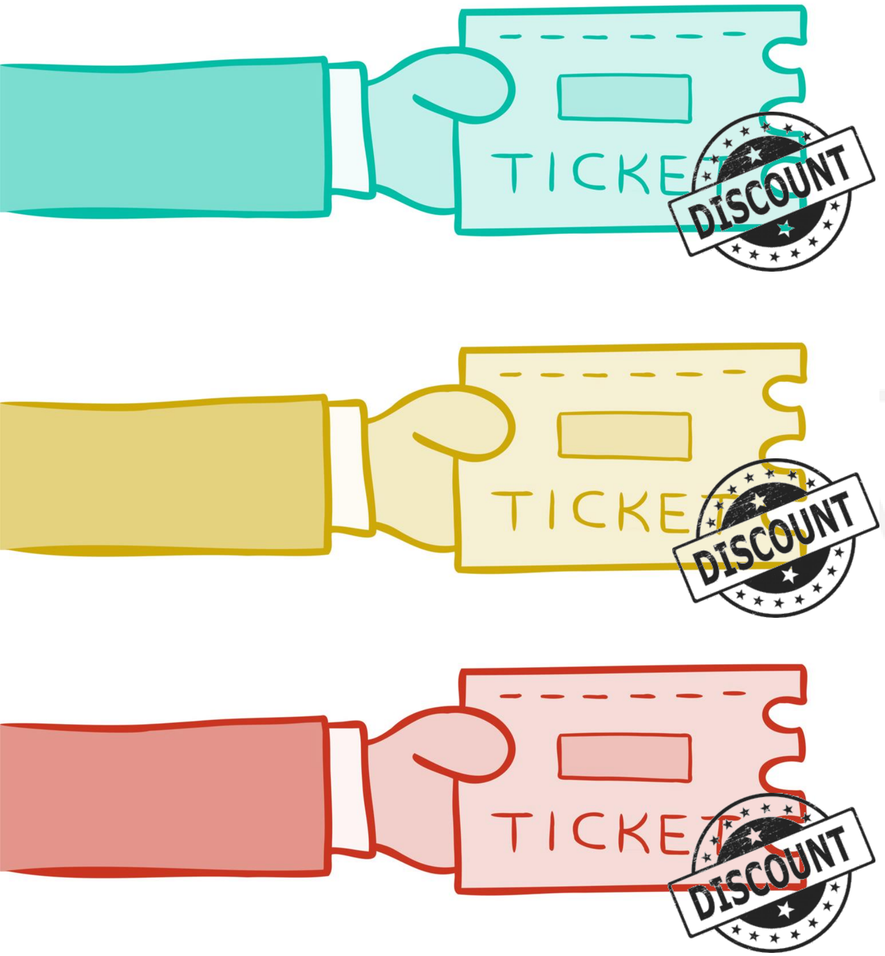
Manipulating arrays



- **forEach:**

```
let studentTickets = [10, 20, 30];

studentTickets.forEach(function(element) {
  console.log(element);
});
```

Manipulating arrays



- **map:**

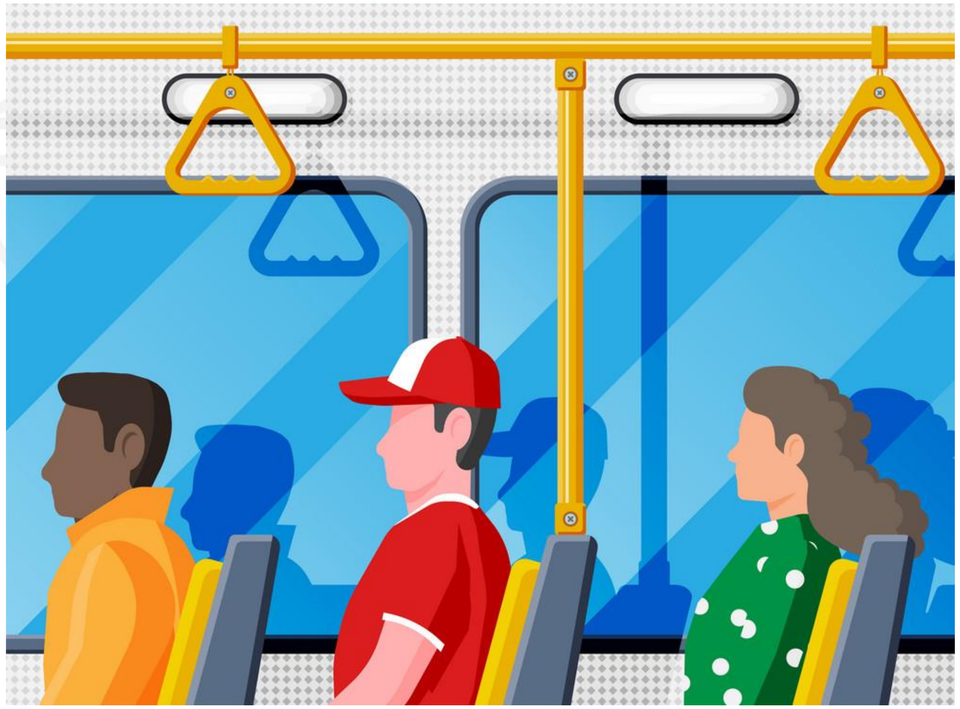
```
const studentTickets = [50, 30, 25, 40];

const newPrice = studentTickets.map(discountFunction)

function discountFunction(num) {
  var amount= num /100*15;
  return num- amount;
}

console.log('MAP - Get array of price after discount 15%');

console.log(newPrice);
```

Manipulating arrays

- **filter:**

```
const allMaleStudents = students.filter(

(std) => std.gender === "male"

);

console.log("FILTER - Get all male students");

console.log(allMaleStudents);

console.log(allMaleStudents.map((s)=> s.name))
```

```
const students = [
    {                   {                   {
        name: 'Majid',      name: 'Ahmed',      name: 'Noor',
        height: 172,        height: 202,        height: 150,
        mass: 77,           mass: 136,          mass: 49,
        gender: 'male',     gender: 'male',     gender: 'female',
    },                  },                  }
                                            ];
```



Important Array Methods

- **push():** Add an item to the end of an array.
- **indexOf():** Find the index of an element in an array.
- **slice():** Make a copy of an array.
- **reduce():** executes a reducer function for each value of an array. reduce() returns a single value which is the function's accumulated result.
- **sort():** method sorts the elements of an array.
- **reverse():** changes the sequence of elements of the given array and returns the reverse sequence.
- **find():** returns the value of the array element that passes a test.
- **shift():**remove the first item of an array
- **Math.max and Math.min:** the minimum or maximum element
- **splice():** method adds and/or removes array elements.

Concatenating Arrays

```
var array1 = [1, 2];

var array2 = [3, 4, 5];

var array3 = array1.concat(array2); //
returns a new array

var array3 = [...array1, ...array2]

Results in a new Array:

[1, 2, 3, 4, 5]
```

Array destructuring

- An array can be destructured when being assigned to a new variable.

```
const triangle = [3, 4, 5];

const [length, height, hypotenuse] = triangle;

length === 3; // → true

height === 4; // → true

hypotneuse === 5; // → true
```

- Elements can be skipped

```
const [,b,,c] = [1, 2, 3, 4];

console.log(b, c); // → 2, 4
```

- An array can also be destructured if it's an argument to a function.

```
function area([length, height]) {

return (length * height) / 2;

}

const triangle = [3, 4, 5];

area(triangle); // → 6
```

Sets

- Like an array in that it stores a collection of items
- Different from an array:
  - Stores only unique/distinct items. A set will, therefore, not store any duplicate values
  - Not indexed
  - Items in a set can’t be accessed individually

```
let myArr = [1, 2, 2, 3, 4, 4, 5, 6, 6];
let arrSet = new Set(myArr);
console.log(arrSet); // Set { 1, 2, 3, 4, 5, 6 }

let myStr = "hello";
let strSet = new Set(myStr);
console.log(strSet); // Set { 'h', 'e', 'l', 'o' }

let arr2 = ["hello", "hello", 2, 2, [1], 1, [3], [3], { a: 1 }, { a: 1 }];
let set2 = new Set(arr2);
console.log(set2);
// Set { 'hello', 2, [ 1 ], 1, [ 3 ], [ 3 ], { a: 1 }, { a: 1 } }
```

Resources

- Can JavaScript Arrays Contain Different Types?
- Work with JavaScript arrays like a boss
- JavaScript Arrays