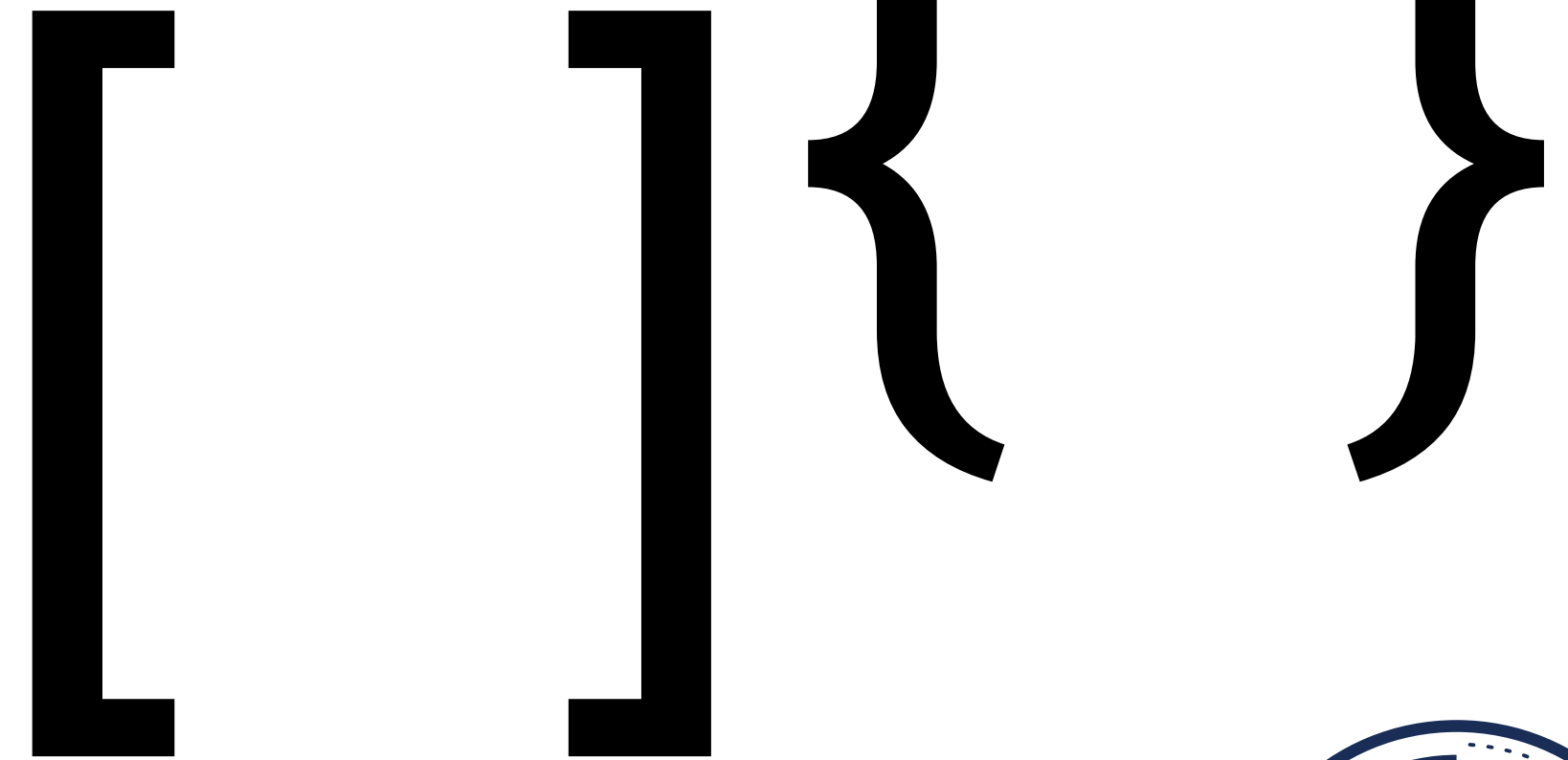


JavaScript essentials for Node-RED

Section 2

Arrays and Objects





What you will learn in this section?

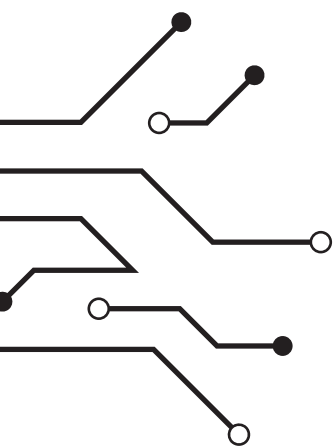
Objects and arrays are used to store multiple values unlike using variables that we used in the last section. Arrays and objects can store multiple values in different data types

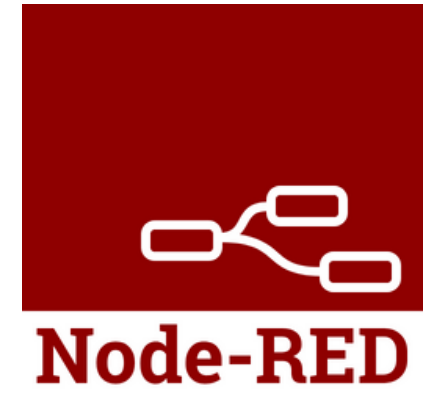
[]

Single and multi-dimensional arrays,
Array methods

{ }

Objects in
JavaScripts





Arrays []

Arrays are basically type of object where you can store multiple values. These multiple values are called as array's element. The elements could have same or different data types

Array element with same data type:

[45, 22, 45, 56]

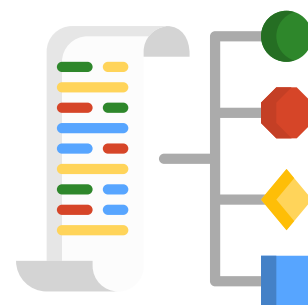
Array element with different data type:

[Rajvir, 22, Germany, Code and Compile]

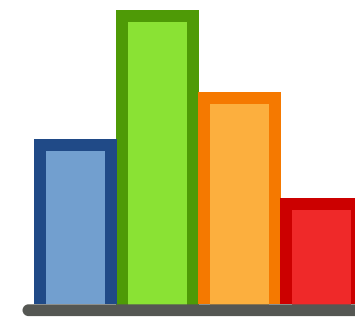
Applications



Data storage and
retrieval

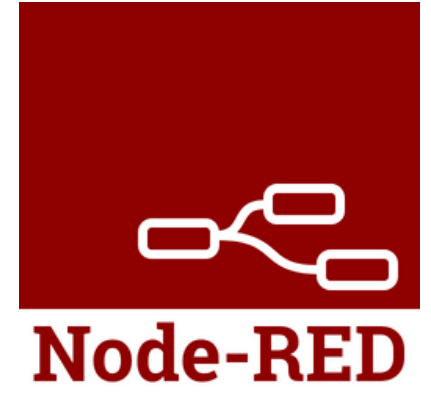


Data sorting and
searching



Graphs and tree
data structure





Single dimension array

Single dimension array

```
1 var arr = [34,56,11,56]; same data type
```

```
2 console.log(arr); [ 34, 56, 11, 56 ]
```

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile']; different data type
```

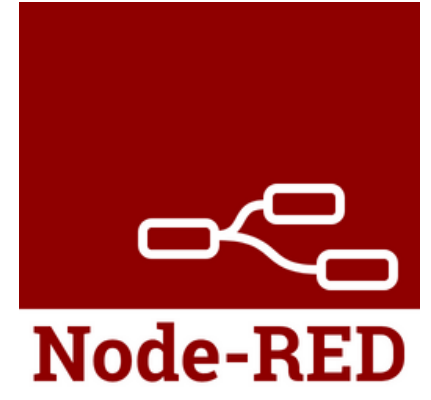
```
2 console.log(arr); [ 'Rajvir', 22, 'Germany', 'Code and Compile' ]
```

Empty array

```
1 var arr2 = new Array(10);
```

```
2 console.log(arr2); [ <10 empty items> ]
```





Accessing array elements

Accessing array elements

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 console.log(arr[0]);  
3 console.log(arr[1]);  
4 console.log(arr[2]);  
5 console.log(arr[3]);  
  
2 console.log(typeof arr[0]);  
3 console.log(typeof arr[1]);
```

Rajvir
22
Germany
Code and Compile

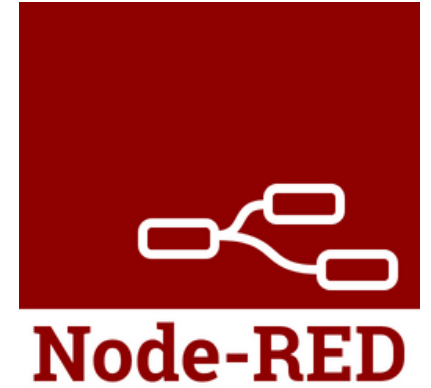
string
number

Accessing array elements out of the index range

```
6 console.log(arr[-1]);  
7 console.log(arr[4]);
```

undefined
undefined





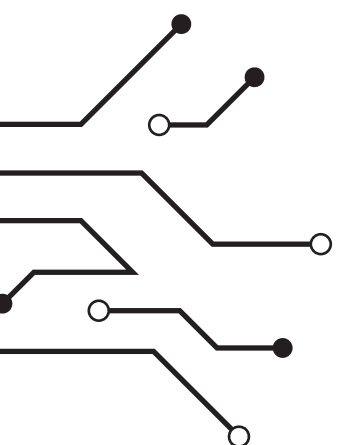
Overwriting array elements

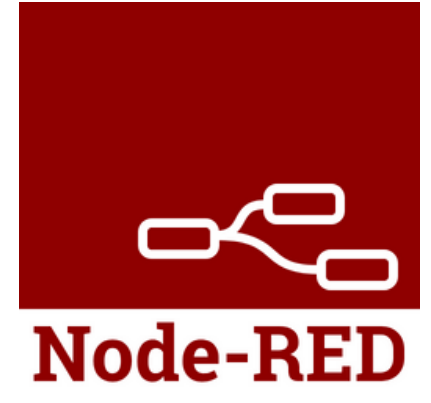
```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];
2 arr[0] = 'Singh';
3 arr[2] = 'Madeira';
4 console.log(arr); [ 'Singh', 22, 'Madeira', 'Code and Compile' ]
```

Writing to elements that do not exist!

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];
2 arr[0] = 'Singh';
3 arr[2] = 'Madeira';
4 arr[-1] = 'JavaScript';
5 arr[4] = 'Node-RED';
6 console.log(arr);
```

```
[
  'Singh',
  22,
  'Madeira',
  'Code and Compile',
  'Node-RED',
  '-1': 'JavaScript'
]
```





Array properties

Array has build-in properties called **length**. This will gives you the number of elements of the array

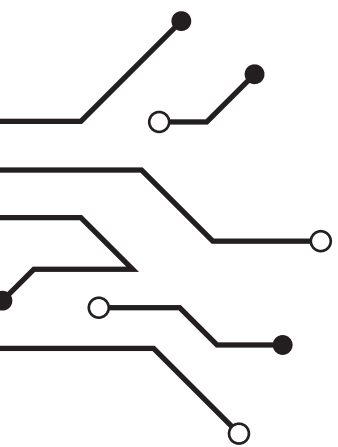
```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 console.log(arr.length); 4
```

```
1 var arr = [];  
2 console.log(arr.length); 0
```

Array length = Max. index - 1

Using array length to access the last element of the array.

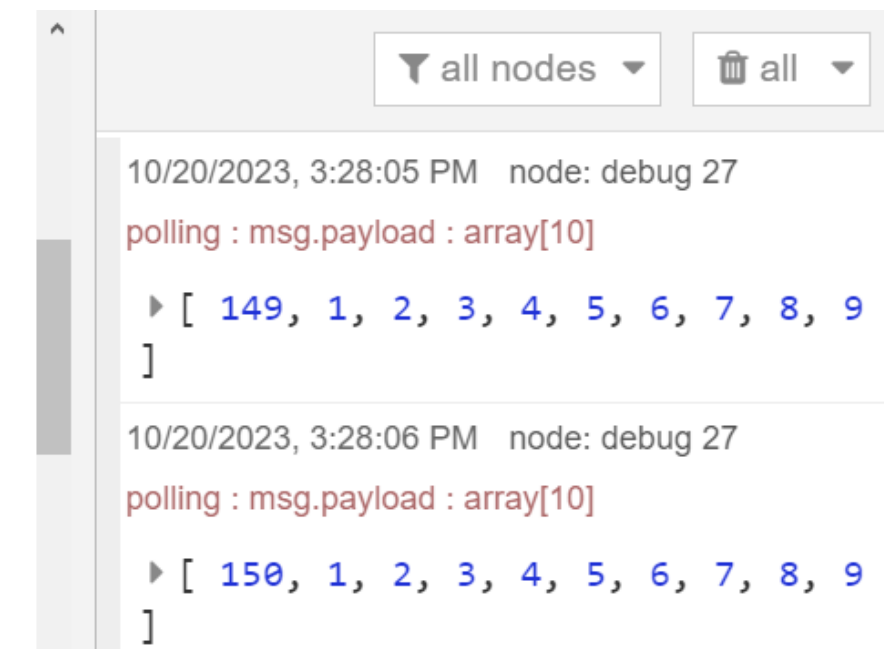
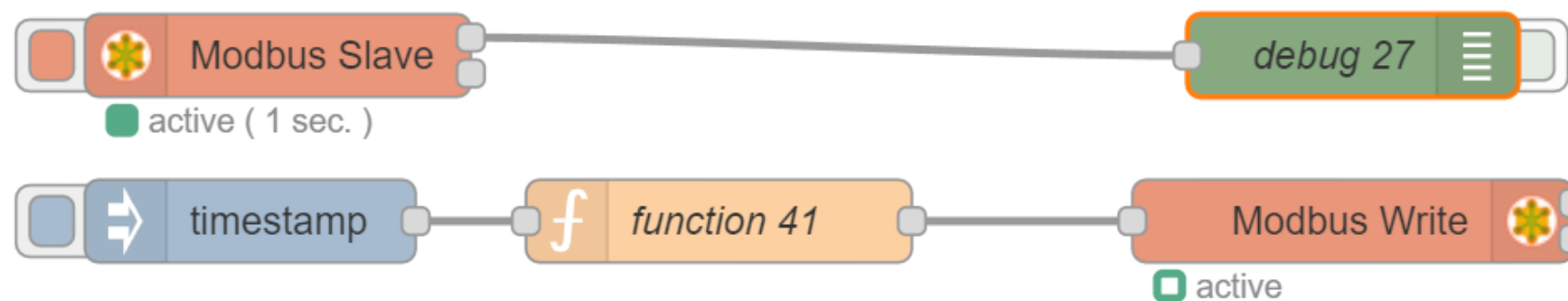
```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 console.log(arr[arr.length - 1]); Code and Compile
```





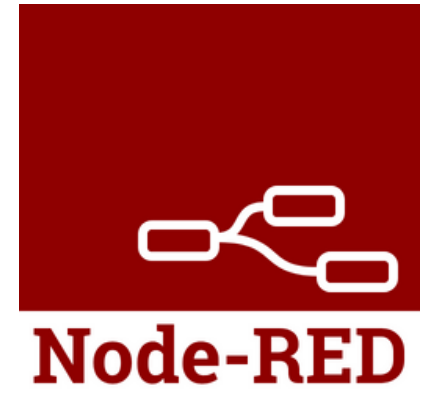
Practice Exercise

Read the data from Modbus Server (PLC or Local)



- Read the status of specific element of the array and display on the dashboard
 - Boolean status (as LED)
 - Integer status (as gauge)
- Write/update the value of the specific element of the array. Change the value using input button or numeric field.



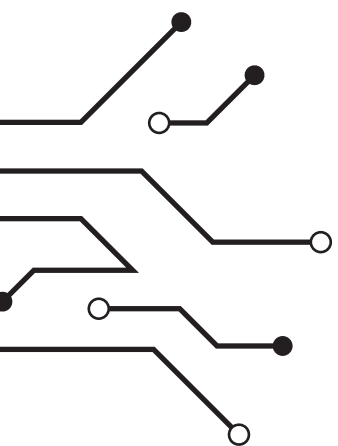


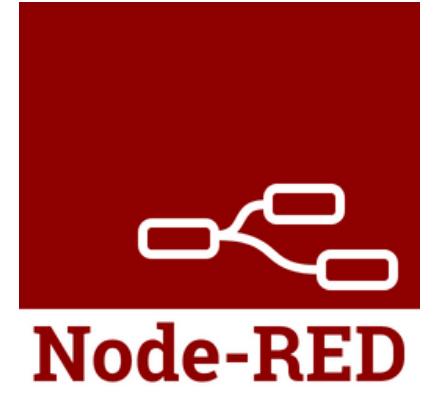
Array properties

Writing to array element that does not exist!

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 arr[5] = '88662'  
3 console.log(arr.length);  
4 console.log(arr);
```

```
6  
[  
  'Rajvir',  
  22,  
  'Germany',  
  'Code and Compile',  
  <1 empty item>,  
  '88662'  
]
```





Array methods

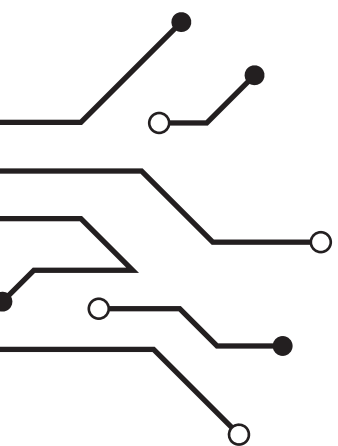
Array methods are used to perform actions on the array. Method is like a function with some code that get execute on the array. The following are some examples:

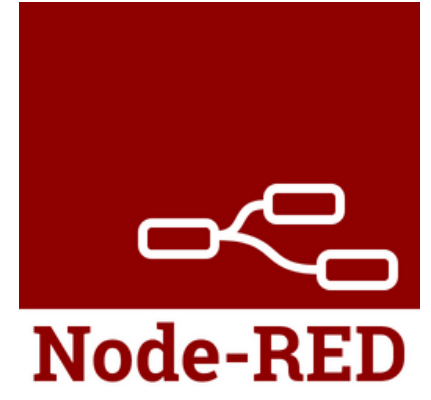
push(): Adding element

```
1 var arr = ['Rajvir', 22];  
2 arr.push('Germany');  
3 console.log(arr); [ 'Rajvir', 22, 'Germany' ]
```

splice(): Adding and replacing element to certain index

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 arr.splice(2,0,'Madeira');  
3 console.log(arr);  
[ 'Rajvir', 22, 'Madeira', 'Germany', 'Code and Compile' ]
```





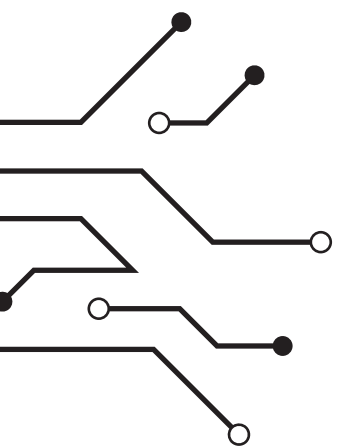
Array methods

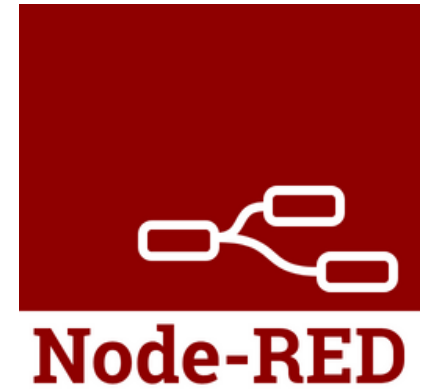
splice(): Adding, replacing or deleting elements to certain index

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];
2 arr.splice(2,1,'Madeira');
3 console.log(arr); [ 'Rajvir', 22, 'Madeira', 'Code and Compile' ]
```

```
1 var arr = ['Rajvir', 22, 'Germany', 'Code and Compile'];
2 arr.splice(2,2,'Madeira');
3 console.log(arr); [ 'Rajvir', 22, 'Madeira' ]
```

```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile'];
2 arr1.splice(2,2);
3 console.log(arr1); [ 'Rajvir', 22 ]
```



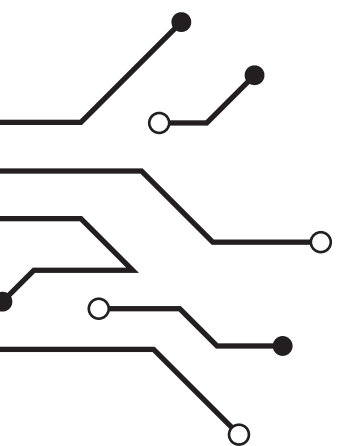


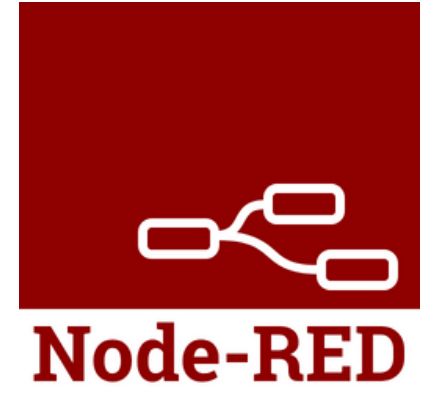
Array methods

concat(): Adding arrays

```
1 var arr1 = ['Rajvir', 22];  
2 var arr2 = ['Germany', 'Code and Compile']  
3 var arr3 = arr1.concat(arr2);  
4 console.log(arr3); [ 'Rajvir', 22, 'Germany', 'Code and Compile' ]
```

```
1 var arr1 = ['Rajvir', 22];  
2 var arr2 = ['Germany', 'Code and Compile']  
3 var arr3 = arr1.concat('Überlingen', '88662');  
4 console.log(arr3); [ 'Rajvir', 22, 'Überlingen', '88662' ]
```





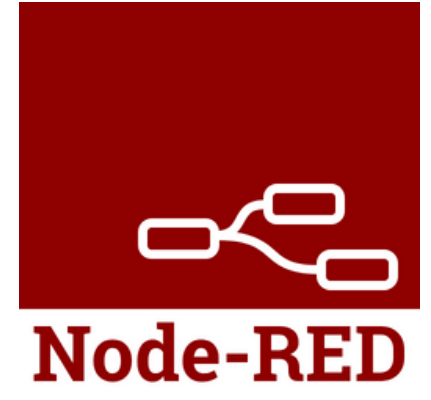
Array methods

pop(): Deleting last elements

```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 arr1.pop(); //removing last element  
3 console.log(arr1); [ 'Rajvir', 22, 'Germany' ]
```

shift(): Deleting first element

```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 arr1.shift(); //removing first element  
3 console.log(arr1); [ 22, 'Germany', 'Code and Compile' ]
```



Array methods

find(): finding elements

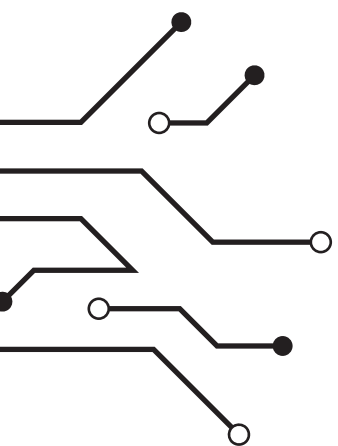
This method can be used to check if the value is present in the array.

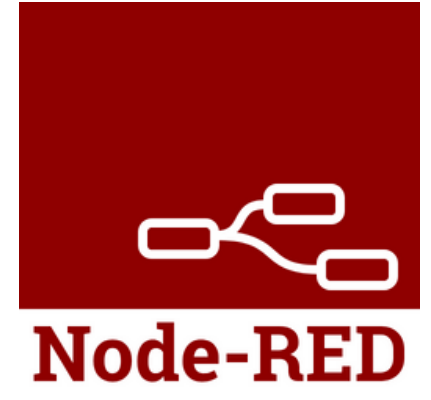
```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 var findValue = arr1.indexOf(arr1.find(e => e === 'Germany'));  
3 console.log(findValue); 2
```

```
2 var findValue = arr1.find(e => e !== 22);  
3 console.log(findValue); Rajvir
```

The method **returns 'undefined'** if the **value does not exist** in the array

```
2 var findValue = arr1.find(e => e === 'Madeira');  
3 console.log(findValue); undefined
```





Array methods

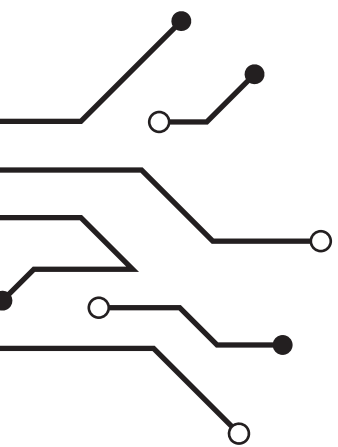
indexOf(): return the index of the elements

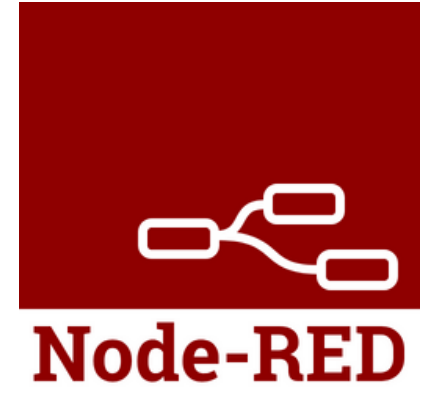
This method is used to return the index number of the element. If the value occurs in the array more than once, it will return the first occurrence.

```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 var index = arr1.indexOf(22);  
3 console.log(index); 1  
  
2 var index = arr1.indexOf(22, 2); //starting from index 2  
3 console.log(index); -1
```

lastIndexOf(): return the last index of the elements

```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile', 22];  
2 var index = arr1.lastIndexOf(22);  
3 console.log(index); 4
```

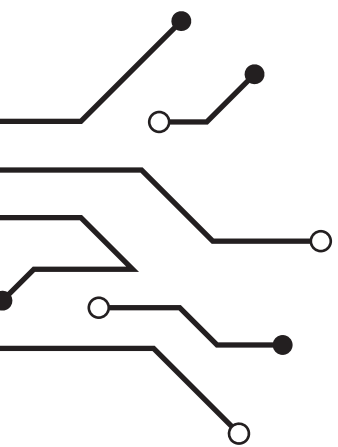


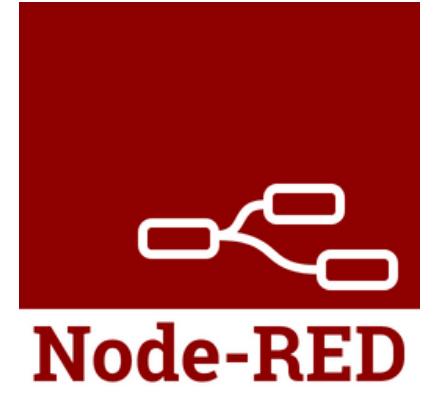


Array methods

Using find() and indexOf() together

```
1 var arr1 = ['Rajvir', 22, 'Germany', 'Code and Compile'];  
2 var findValue = arr1.indexOf(arr1.find(e => e === 'Germany'));  
3 console.log(findValue); 2
```





Array methods

String Sorting

Sorting

It sorts the array elements based on the datatype of the elements:

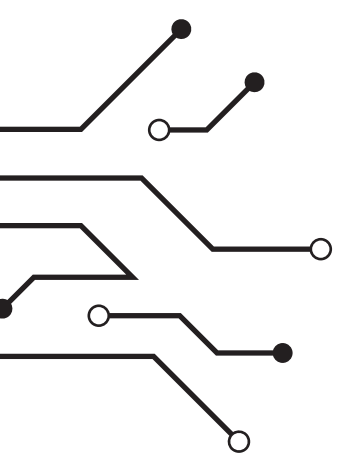
- **Alphabets: A-Z**

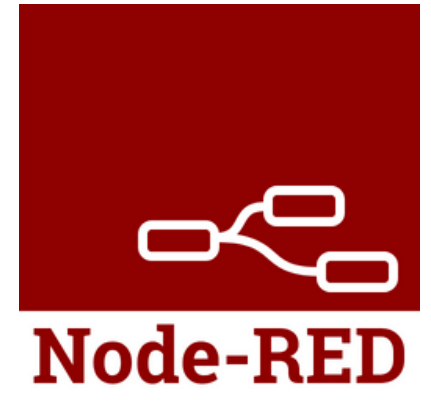
```
1 var arr1 = ['Rajvir', 'Germany', 'Code and Compile'];  
2 console.log(arr1.sort()); [ 'Code and Compile', 'Germany', 'Rajvir' ]
```

Reversing

The elements of array can be reversed using this function

```
1 var arr1 = ['Rajvir', 'Germany', 'Code and Compile' ];  
2 console.log(arr1.reverse()); [ 'Code and Compile', 'Germany', 'Rajvir' ]
```





Array methods

Numeric Sorting

Sorting in Ascending order

```
1 var arr = [40, 100, 1, -6, -7, 5, 25, 10];  
2 msg.payload = arr.sort(function(a, b){return a-b});  
3 return msg;
```

msg.payload : array[8]

► [-7, -6, 1, 5, 10, 25, 40, 100]

Alternate way to write the function

```
2 msg.payload = arr.sort((a, b) => {return a-b});
```

Sorting in Descending order

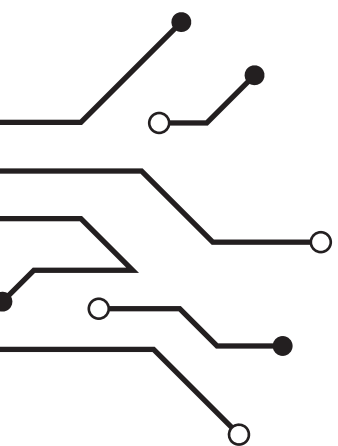
```
1 var arr = [40, 100, 1, -6, -7, 5, 25, 10];  
2 msg.payload = arr.sort((a, b) => {return b-a});  
3 return msg;
```

msg.payload : array[8]

► [100, 40, 25, 10, 5, 1, -6, -7]

Alternate way to sort in descending order

```
2 msg.payload = arr.sort(function(a, b){return a - b}).reverse();
```



Array methods

Numeric Sorting using TypedArray

Sorting in Ascending order

```
1 var arr = new Int16Array([40, 100, 1, -6, -7, 5, 25, 10]);
2 msg.payload = arr.sort();
3 return msg;
```

Sorting in Descending order

```
1 var arr = new Int16Array([40, 100, 1, -6, -7, 5, 25, 10]);
2 msg.payload = arr.sort().reverse();
3 return msg;
```



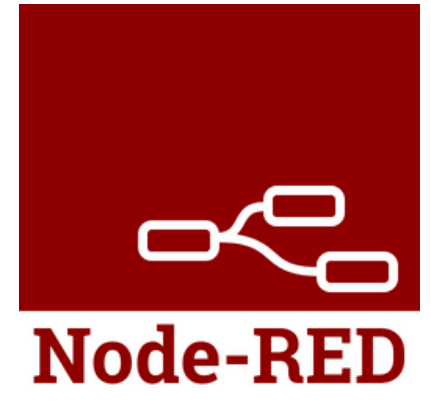
We will learn later how to convert the Object back to array



Ascending
order

▼ payload: object

```
0: -7
1: -6
2: 1
3: 5
4: 10
5: 25
6: 40
7: 100
```

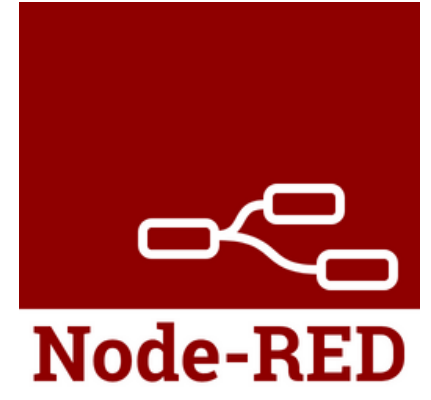


Descending
order

▼ payload: object

```
0: 100
1: 40
2: 25
3: 10
4: 5
5: 1
6: -6
7: -7
topic: ""
```





Array methods

Finding minimum and maximum value in an Array

There are no built-in functions for finding the max or min value in an array. However, after you have sorted an array, you can use the index to obtain the highest and lowest values.

Minimum value

```
1 var arr = [40, 100, 1, -6, -7, 5, 25, 10];
2 var sort = arr.sort(function (a, b) { return a - b });
3 msg.payload = sort[0];
4 return msg;
```

Max value

```
1 var arr = [40, 100, 1, -6, -7, 5, 25, 10];
2 var sort = arr.sort(function (a, b) { return a - b });
3 msg.payload = sort[sort.length - 1];
4 return msg;
```

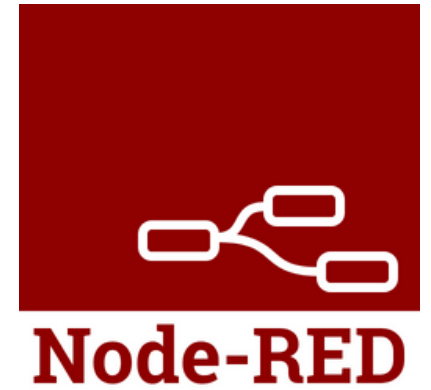


Sorting a whole array is a very inefficient method if you only want to find the highest (or lowest) value.



We will learn alternate way to find Max and Min later in the course when we talk about functions

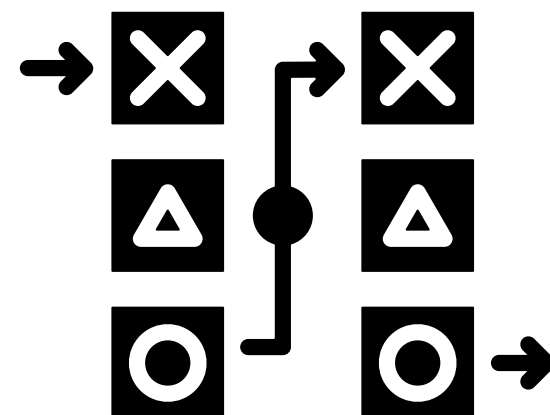




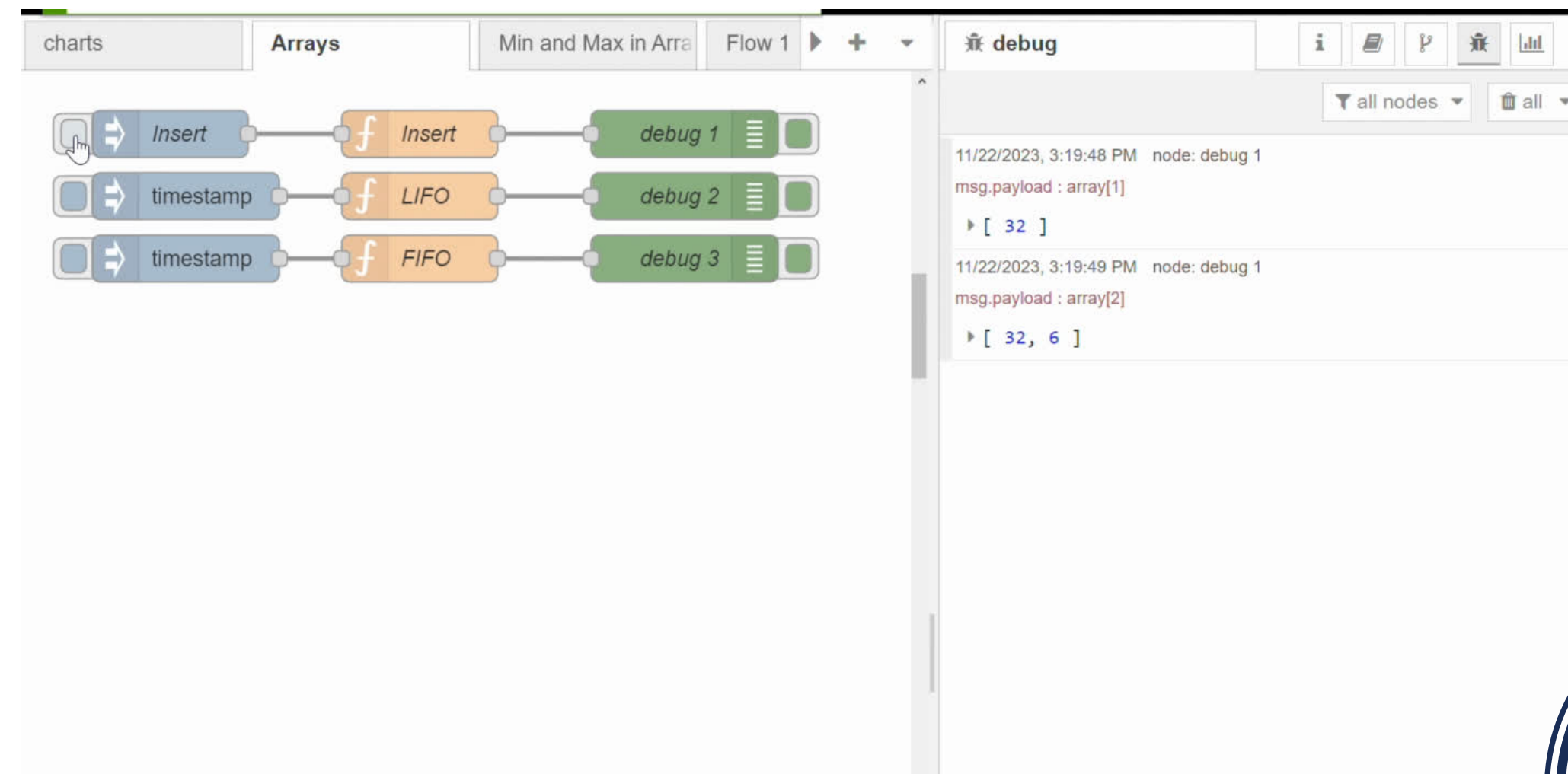
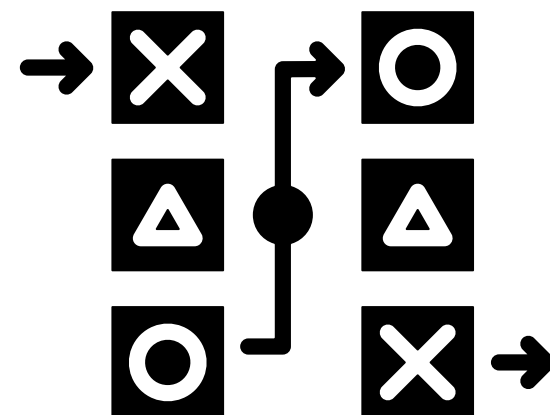
Practice Exercise

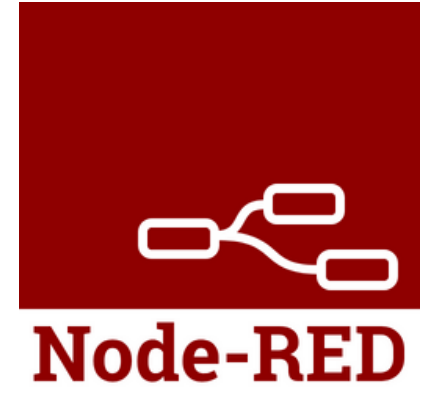
Create the following methods using array methods:

FIFO (First-in-First-out)



LIFO (Last-in-First-out)





Multi-dimensional array

These are basically array of an array

Two-dimensional array

```
1 var arr1 = [1,2,3];  
2 var arr2 = [4,5,6];  
3 var arr3 = [7,8,9];  
4 var multiArray = [arr1,arr2,arr3];  
5 console.log(multiArray);
```

```
[ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ]
```

Accessing element

```
5 console.log(multiArray[1][1]);
```



Multi-dimensional array

Three-dimensional array

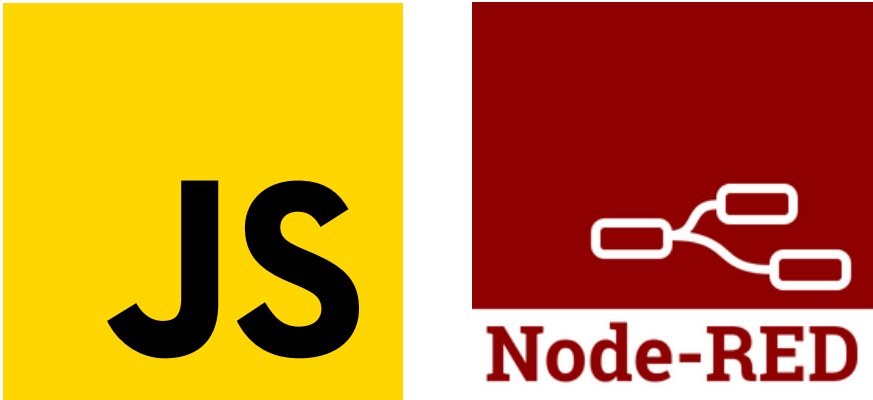
```
1 var arr1 = [1,2,3];
2 var arr2 = [4,5,6];
3 var arr3 = [7,8,9];
4 var multiArray = [arr1,arr2,arr3];
5 var multiArray2 = [multiArray, multiArray, multiArray]
6 console.log(multiArray2);
```

```
[
  [ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ],
  [ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ],
  [ [ 1, 2, 3 ], [ 4, 5, 6 ], [ 7, 8, 9 ] ]
]
```

Accessing element

```
6 console.log(multiArray2[2][2][1]);
```

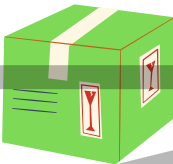


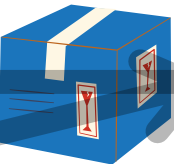

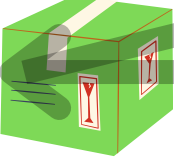
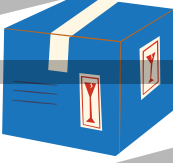

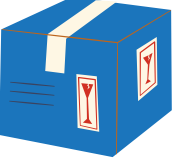

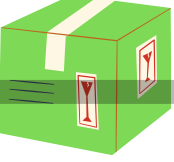
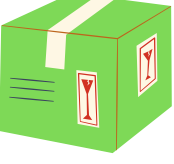




Project 1

Store the product information in the array where we use the value of row and column to find out the product type stored in the array. Use methods to store and remove the product from the array



11	← 3,2	 3,1	 3,0	9	Product type	Product Nr.
8	 2,2	2,1	 2,0	6		3
5	 1,2	 1,1	 1,0	3		2
2	← 0,2	 0,1	 0,0	0		1
					Empty	0

