```
/*
Array is a object that accepts more than value in a single variable name.
Array members are indexed. The first member has an index of 0, second member has
an index of 1,....
There different types of arrays
1. dense array
2. sparse array
There are different ways of using(declaring) array
1. Using array literal
2. Using the new keyword
*/
// Using the array literal
const fruits = ["Apple", "Orange", "Banana", "Cherry"];
console.log(fruits);
// To access individual members of the array, we use the index in a square bracket
console.log(fruits[0]);
console.log(fruits[3]);
// Array accept any data type
const data = [
  40,
  "Hello",
  true,
  false,
  { name: "Mary Doe", email: "marydoe@gmail.com" },
  ["Uche", "Ade", "Lawal"],
];
```

```
console.log(data);

console.log(data[5][2]);

console.log(data[0].name);

const person = { name: "Mary Doe", email: "marydoe@gmail.com" };

// console.log(person["email"]);

// Checking the data type of an array

console.log(typeof data);

console.log(Array.isArray(data));

console.log(Array.isArray(person));

// Checking the length of an array

console.log(data.length);

// Creating of an array using the new keyword

const arr = new Array("Book", "Pencil", "Rular", "Pen");

console.log(arr);

console.log(arr[1]);

// Showing the difference between array literal and the new keyword

const arr1 = [10];

const arr2 = new Array(10);

console.log(arr1[0]);

console.log(arr2[9]);

const arr3 = [2, , 4, , 5, , 6];

console.log(arr3[1]);

// Updating an array

arr3[0] = 7;

arr3[1] = 3;

arr3[3] = 8;
```

```
delete arr3[4];

delete arr3[5];

console.log(arr3);

// Looping through the members of the array using for loop

/*

for(initialization;condition;increment/decrement){

// do something(body)

}

*/

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

  console.log(arr[i]);

}


for (let i = arr.length - 1; i >= 0; i--) {

  console.log(arr[i]);

}

// Built in Array methods

// pop(), push(), shift(), unshift()|

// stack: LIFO(Last In First Out); FIFO(First In First Out)

const stack = ["a", "b", "c", "d"];

console.log(stack);

stack.pop();

console.log(stack);

stack.push("d");

console.log(stack);

stack.shift();
```

```
console.log(stack);

stack.unshift("a");

// slice(), splice()

/*

slice(start,end): This helps us to get a fraction of an array

The start talks about the index at which you want to start cuttting the array

The end talks about where you want to end. And it is optional

Note that the end value is not included rather the index immediately before the end

*/

console.log(stack.slice(2));

console.log(stack.slice(1, 3));

// splice(): This is used to insert items into an array and also to remove items from

// the array. splice(start,number of elements to be removed, items to be added)

stack.splice(4, 0, "e", "f", "g");

console.log(stack);

stack.splice(1, 2, "r");

console.log(stack);

// forEach(), map(), filter(), find(), findIndex()

/*

array.forEach(a callback function(v,i,a))

The callback function(v,i,a): Is a function that runs on each member of the array

where v = the array elements(array members), i = the array index, a= the array itself

*/

const num = [1, 2, 3, 4, 5, 6, 7];

num.forEach(function (i) {

  console.log(i ** 2);
```

```
});

// map() const num = [1, 2, 3, 4, 5, 6, 7];

const x = num.map((v) => {

  return v ** 3;

});

console.log(x);

// filter() const num = [1, 2, 3, 4, 5, 6, 7];

// filter accepts a conditional callback function. And returns all the elements that

// meets that condition

const even = num.filter((v) => {

  return v % 2 == 0;

});

console.log(even);

const odd = num.filter((v) => {

  return v % 2 != 0;

});

console.log(odd);

const e = num.filter((i) => {

  return i > 4;

});

console.log(e);

// find() const num = [1, 2, 3, 4, 5, 6, 7];

// find() accepts a conditional callback function and returns the first element

// that meet the condition

const t = num.find((v) => {

  return v > 4;
```

```
});

console.log(t);

//findIndex() const num = [1, 2, 3, 4, 5, 6, 7];

// The find() method returns the element itself while the findIndex() returns the

// index of the array element

const r = num.findIndex((v) => {

  return v > 4;

});

console.log(r);
```