```
/*
A loop is a block of code that runs as long as a certain condition is true.
There are different types of loop in JavaScript
1. for loop
2. while loop
3. do while loop
4. for in loop
5. for of loop
6. forEach(), map(), filter()
syntax:
for(intialization;condition; increment/decrement){
// block of code that does sth
}
*/
const fruits = ["Apple", "Banana", "Cherry", "Orange"];
console.log(fruits[0]);
console.log(fruits.length);
// Loop with incrementor
for (let a = 0; a < fruits.length; a++) {
  console.log(fruits[a]);
}
// Loop with decrementor
for (let i = fruits.length - 1; i >= 0; i--) {
  console.log(fruits[i]);
}
```

```
// Strings are seen as arrays in javascript
const name = "John Doe";
for (let i = 0; i < name.length; i++) {
  console.log(name[i]);
}

// civic, level, racecar
const pal = function (n) {
  let text = "";
  for (let i = n.length - 1; i >= 0; i--) {
    text += n[i];
  }
  if (text == n) {
    return `${n} is a palindrome`;
  } else {
    return `${n} is not a palindrome`;
  }
};
console.log(pal("mary"));
// The while loop
/*
syntax:
initialization
while(condition){
// do sth
increment/decrement
}
const fruits = ["Apple", "Banana", "Cherry", "Orange"];
*/
let i = 0;
while (i < fruits.length) {
  console.log(fruits[i]);
  i++;
}
// Looping a text array using the decrementor
lastName = "Ademide";
let em = "";
let x = lastName.length - 1;
while (x >= 0) {
  console.log(lastName[x]);
  x--;
}

// lastName = "Ademide";
// let em = "";
// let x = lastName.length - 1;
// while (x >= 0) {
//   em += lastName[x];
//   x--;
// }
// console.log(em);
// The do while loop const fruits = ["Apple", "Banana", "Cherry", "Orange"];
/*
initialization
do{
// do sth
increment/decrement
}while(condition)
*/
let w = 0;
do {
  console.log(fruits[w]);
  w++;
} while (w < fruits.length);

// The for in loop. This is usually used to iterate through an object
/*
syntax:
for(let a in object){
// do sth
}
In this case,'a' is a  local variable that holds the keys of the object
*/
const person = {
  name: "Lawal",
  age: 20,
  email: "lawal@gmail.com",
};
for (let i in person) {
  console.log(person[i]);
}
// Nested Object
const parent = {
  child1: {
    name: "Doe Little",
    email: "doelittle@gmail.com",
  },
  child2: {
    name: "Doe Little 2",
    email: "doelittle2@gmail.com",
  },
  child3: {
    name: "Doe Little 3",
    email: "doelittle3@gmail.com",
  },
};
// How to get a particular value directly
console.log(parent["child2"]["name"]);
for (let i in parent) {
  //   console.log(parent[i]);
  for (let j in parent[i]) {
    console.log(`${j}:${parent[i][j]}`);
  }
```

```
}
console.log(parent["child2"]["name"]);
console.log(parent.child2.email);
// The for of loop: This also comes with a local variable just like the for in loop
// Just that the local variable holds the array elements as against the keys which
// was the case in the for in loop
const food = ["Yam", "Garri", "Rice", "Beans"];
for (let i of food) {
  console.log(i);
}
// Breaking out of a loop Or skipping a particular member
for (let i = 0; i < food.length; i++) {
  if (food[i] === "Garri") {
    break;
  }
  console.log(food[i]);
}
// the continue statement skips a member
for (let i = 0; i < food.length; i++) {
  if (food[i] === "Rice") {
    continue;
  }
  console.log(food[i]);
}

const num1 = 5;
```

```
  },

const num2 = "5";
console.log(num1 === num2);
console.log(typeof num1);
console.log(typeof num2);

// Looping through a nested array
const arr4 = [
  [1, 2],
  [3, 4],
  [5, 6],
];
// console.log(arr4);
let sumation = 0;
for (let i = 0; i < arr4.length; i++) {
  // console.log(arr4[i]);
  for (let j = 0; j < arr4[i].length; j++) {
    sumation += arr4[i][j];
    console.log(arr4[i][j]);
  }
}
console.log(sumation);
const obj = {
  child1: {
    name: "Child One",
    email: "childone@gmail.com",
  },
  child2: {
    name: "Child Two",
    email: "childtwo@gmail.com",
  },
  child3: {
    name: "Child Three",
    email: "childthree@gmail.com",
  },
};

// Looping through a nested object
for (let x in obj) {
  for (let y in obj[x]) {
    console.log(obj[x][y]);
  }
  // console.log(obj[x]);
}
```