**JavaScript:**

  \* JavaScript is a Scripting programming language for Web interactivity.

  \* It is a light Weight programming language. (light weight is a one that's easy to use, doesn't need heavy system source.

  \* It is a loosely typed programming language/ Not strictly typed. (loosely typed- don't have to declare the variable type explicitly).

  \* Main Disadvantages of JavaScript is security because it runs on client side.

  \* JavaScript can be executed inside any browser and it is using JavaScript engines (V8).

  \* JavaScript is a Interpreter programming language.

**What is the Difference between compiler and interpreter?**

**Compiler:**

  \* Compiler converts the entire High-Level Language into machine level language in one shot.

  \* Compilation uses a software called a compiler.

  \* Faster in execution.

  \* Example: C, Java, C++.

**Interpreter:**

  \* Interpreter converts one instruction at a time.

  \* Interpretation uses a software called Interpreter.

  \* Slower in execution.

  \* Example: Python, JavaScript.

**Datatypes in JavaScript:**

In JavaScript, there are 8 main data types, divided into primitive and non-primitive types.

**Primitive:**

  Primitive types hold a single value and are immutable (cannot be changed)

**\* String;** text data (let name ="Kalaivanan";)

  \* **Number;** both integer and float (let age = 25; let price = 99.9)

  \* **BigInt;** used for very large integers beyond the number limit.(let bigNumber = 12345678098762344n;). BigInt can be used only in integer.

  \* **Boolean;** true or false (let isOnline = true;)

  \* **Undefined;** A variable declared but not assigned a value (let x; console.log(x);//undefined)

  \* **Null;** intentional empty (let data = null;)

  \* **Symbol;** to create unique identifiers (let id = Symbol("id"))

**Non-Primitive:**

  Non primitive types hold a multiple value.

**\* Object;** used to store collections of data and complex entities. (let person = {name: "kalai",age: 20};)

**Type Conversion:**

**\*** Type conversion in JavaScript means changing a value from one data type to another data type (example: converting a string to a number or number to a string)

**\*** There are two main Type conversion

  \* Implicit Conversion.

  \* Explicit Conversion.

**Implicit Conversion:**

JavaScript automatically converts types when needed.

  Example:

  // String + Number -> String.

  let result1 = '5' + 2;

  console.log(result1); // '52' (number 2 converted to string)

  // Number - String -> Number;

  let result2 = '5' - 2;

  console.log(result2); // 3 (string '5' converted to number)

**\*** + with a string -> converts everything to string

**\*** Other arithmetic operator (-, \*, /) -> converts to numbers.

**Explicit Conversion:**

  Manually convert a value using built-in functions.

  Example:

  // To string

  String (123); // "123"

  (123). toString (); // "123"

  // To Number

  Number ("123"); //123

  parseInt ("123.45") // 123

  parseFloat ("123.45") // 123.45

**Variable:**

\*Variable is a container that store a data value.

  \* There are some rules to declare the identifier or variable.

  \* Identifier Should Contain only a-z, A-Z, 0-9, \_, $.

  \* Identifier should not start with digits 0-9.

  \* Identifier should not contain any special char/symbol

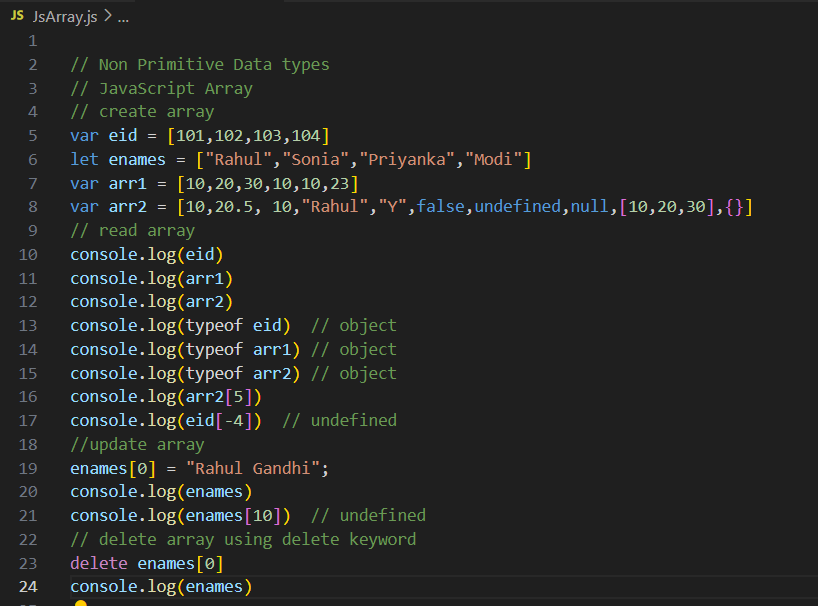
  \* Identifier should not use keyword.

  \* Identifier are case sensitive.

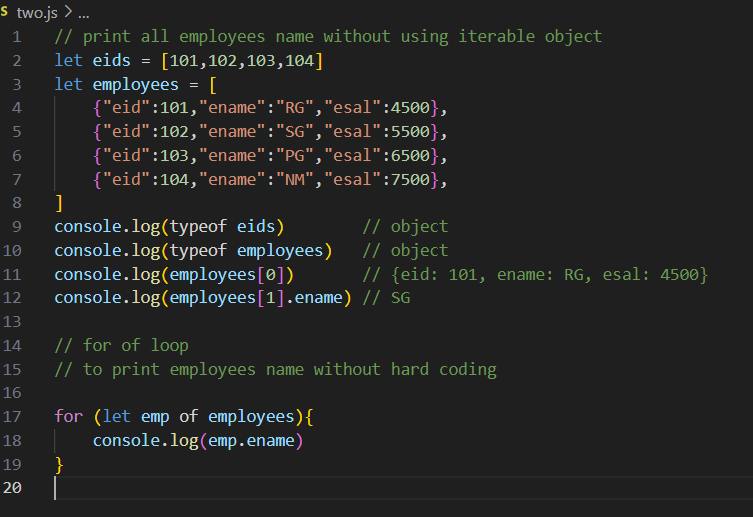
**JavaScript Array:**

* Array is a group of elements as one entity.
* It allowed duplicates and heterogenous. ( heterogenous mean it store different type of data type like (number, string , boolean))
* Array elements are storing based indexing [0 to n-1]
* Negative indexing is not possible
* Array is a iterable object (like for, while, do while loop)

**Example:**



**Example:**



**JavaScript Object:**

* Object is group of key: value pairs as one entity
* Duplicates keys are not allowed.
* Indexing concept is not applicable
* Order no guarentee
* JavaScript object is not iterable

**Example:**



**JavaScript Operator:**

**JavaScript Iterative Statements**

* For Loop
* Whilel Loop
* Do while Loop
* For of loop => []
* For in loop =>{}

**for loop:**

A for loop is used to repeat a block of code specific numbers of times.

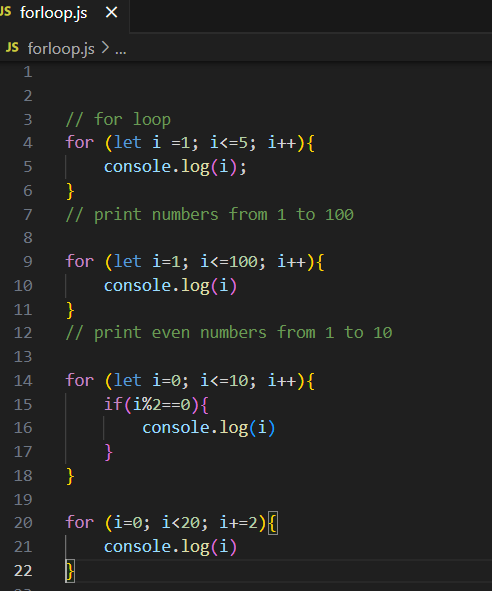
**Syntax:**

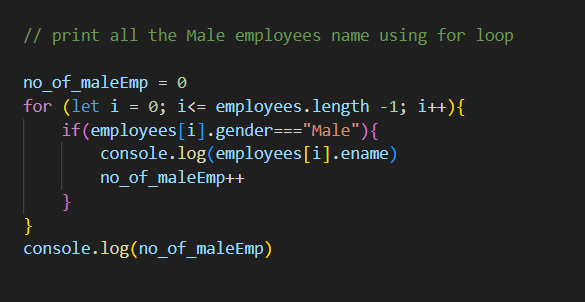
for(initialization, condition, increment/decrement){

// code be executed

}

**Example:**

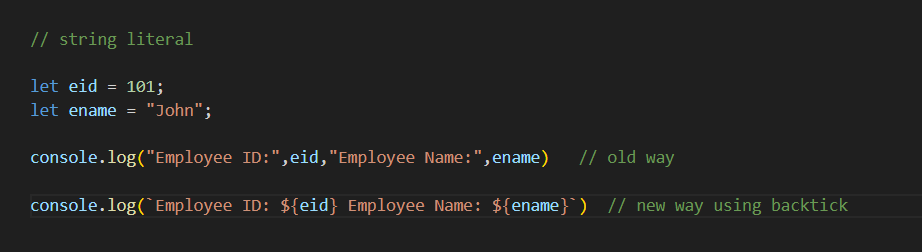


****

**String literal:[ ~${}~]**

* Read variable in the form of string.
* For constructing URL

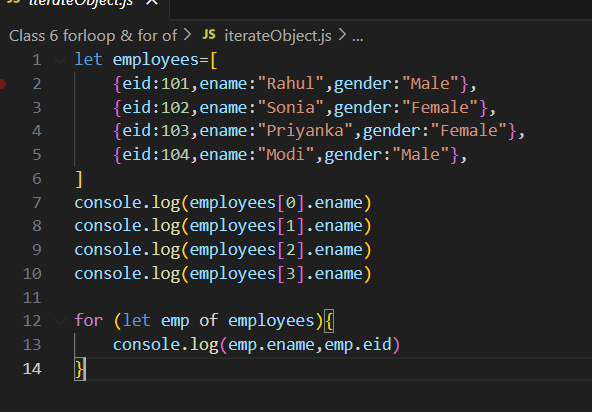
**Example:**

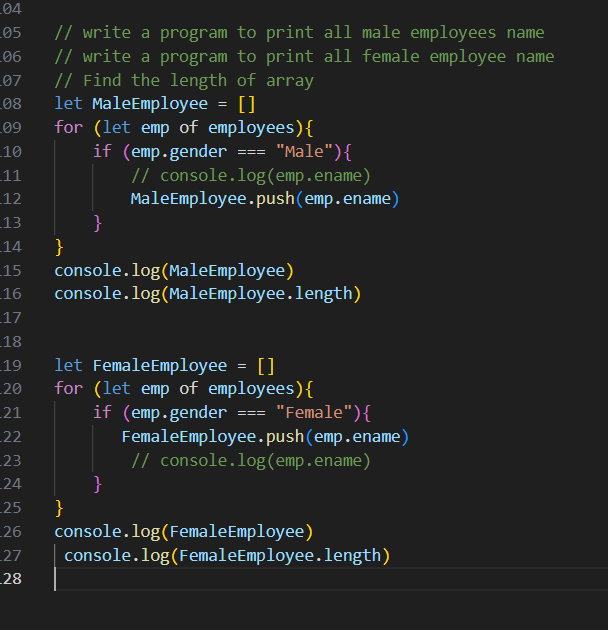


**For of loop:**

The for of loop in JavaScript is used to iterate over the elements of an iterable object – like an array, string, map etc.

**Example:**





**While loop:**

While loop is used to repeat a block of code as long as given condition is true.

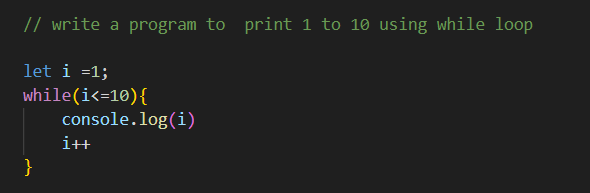
**Syntax:**

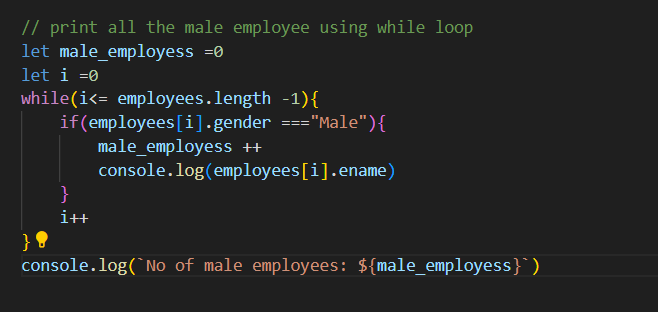
while (condition) {

// code to be executed

}

**Example:**

****

****

**Do While loop:**

It executes the block of code at least once, if the condition is false.

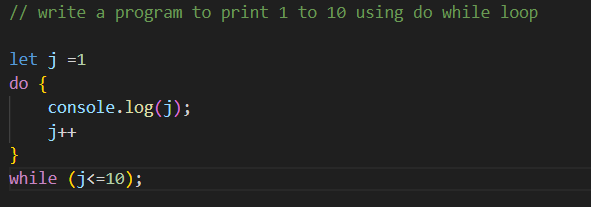
**Syntax:**

do {

// code to be executed

} while (condition);

**Example:**

****

**JavaScript Function:**

* Function is reuseable code to perform a specific task.
* Function can be created using function keyword.

**Syntax:**

Function login () {

Console.log (“login created successfully”)}

Login ()

Return and Break:

* The break statement ends of the loop where it appears and jumps of it
* The return statement comes out from the block or function.

Example:

