LEARN JAVASCRIPT

1. What is javascript

2 . Role of javascript
in web development

3. Variables & Data types

4. Variable mutation &
Type coercion

5. Alert & prompt

6. Basic Operators

7. Operator precedence

8. Control structure &
Statement

9. Ternary operators &
Switch statement

10. falsy & truthy values

11. Equality operators

12. Functions

13. Function statements &
Expressions

14. Arrays

15. Object & Properties

16. Object Methods

17. Loops & iteration

WHAT IS JAVASCRIPT

javascript is a lightweight, cross platform object oriented programming language

1. Lightweight means it doesn't eat much computer memory and it has relatively simple syntax and features

2. Cross platform means language can be use on multiple platforms, systems not just for web development

3. Object-oriented means language that's based on objects

* Together with html & css javascript is the one of the three core technologies of web development.

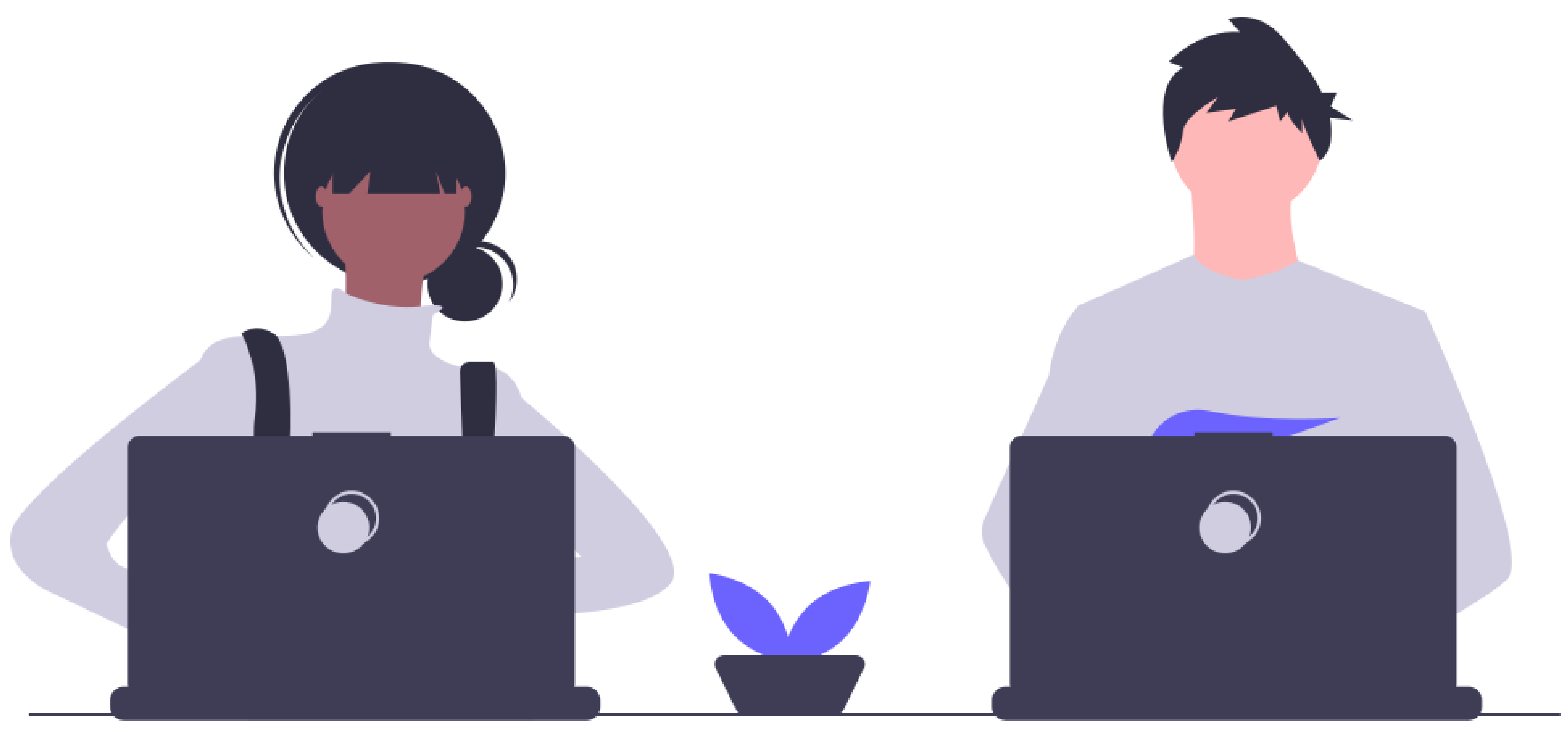
* Today javascript can be used in different places :-
  1) client-side :-
     javascript was traditionally only used in browser

  2) Server-side :-
     Thanks to node js we can use javascript on server as well

rohit_virkud

JAVASCRIPT MAKES MORDEN WEB DEVLOPMENT POSSIBLE

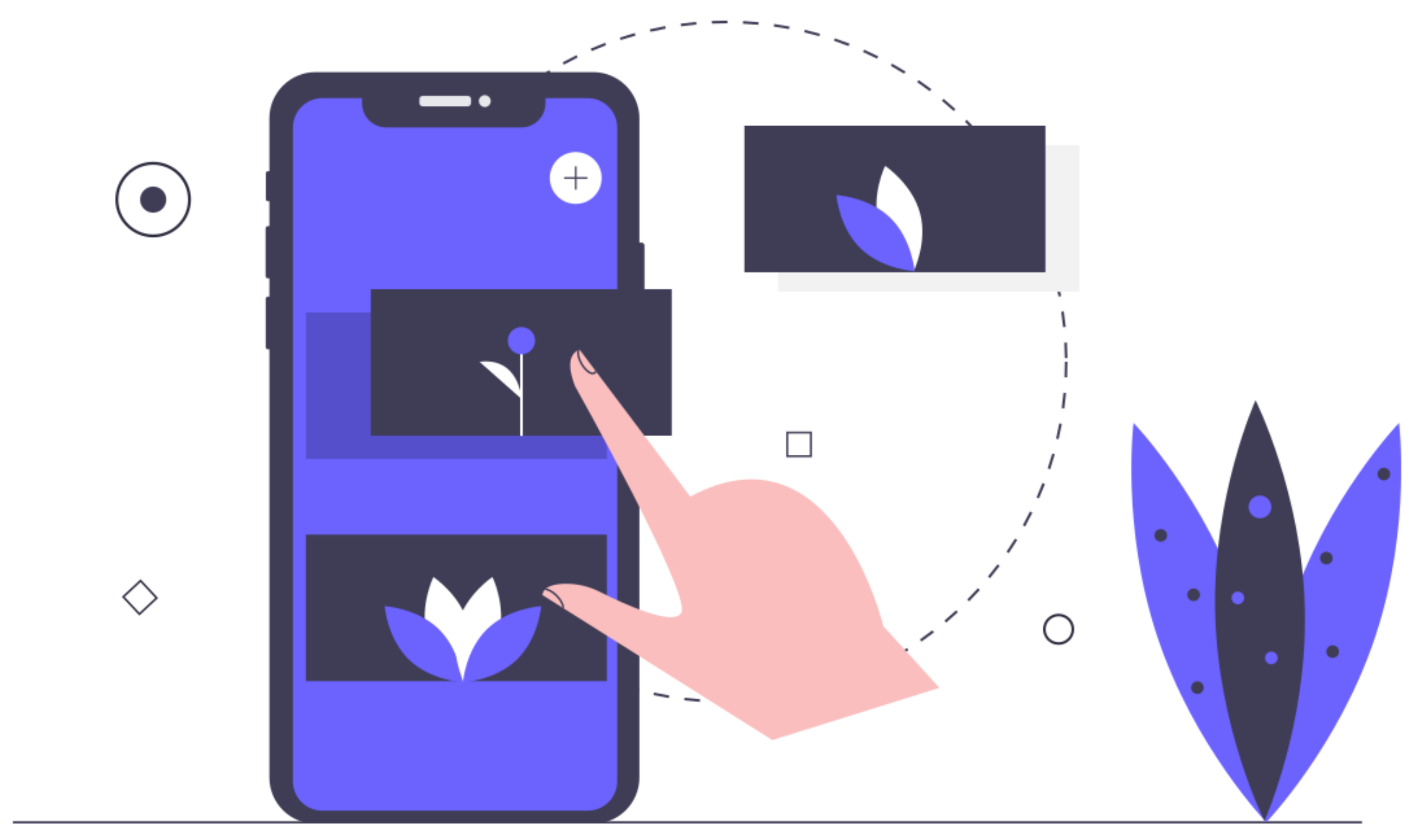
It allows us to not only add dynamic effect and interactivity to more simpler websites but it also allows us to create complex modern web applications that we can interact with that also feels like apps that we use on our computers and phones every single day



FRAMEWORKS :-

Today there are a lot javascript libraries and frameworks out there that implements different architectures and helps developers to build more complex apps more easily and faster, here i’m talking about the stuff like react , angular or even jquery which are all extremely popular and i am sure that you heard of them already right,

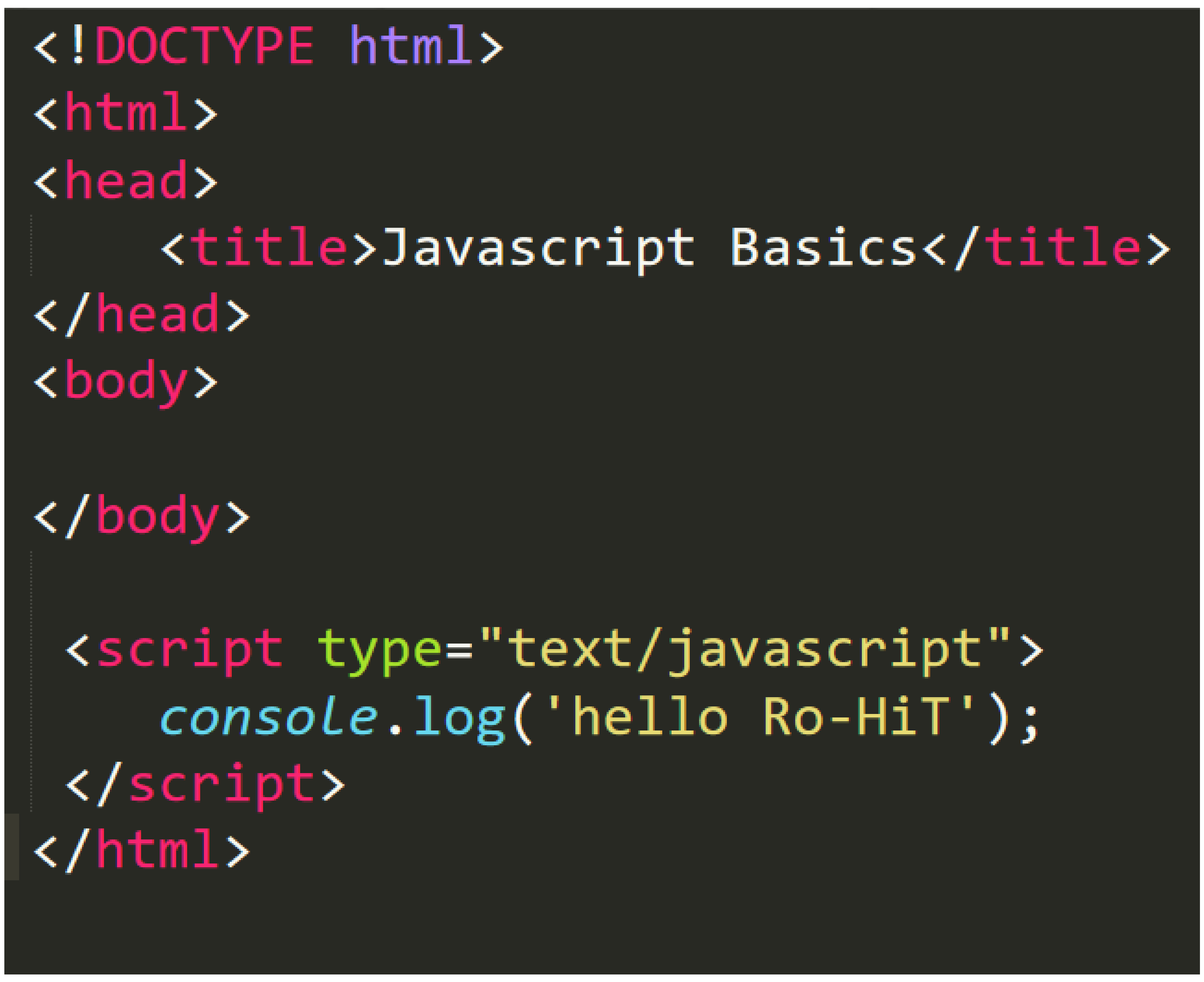
The thing is they are 100% based on javascript so before learning frameworks or using them you have to be really good in javascript



rohit_virkud

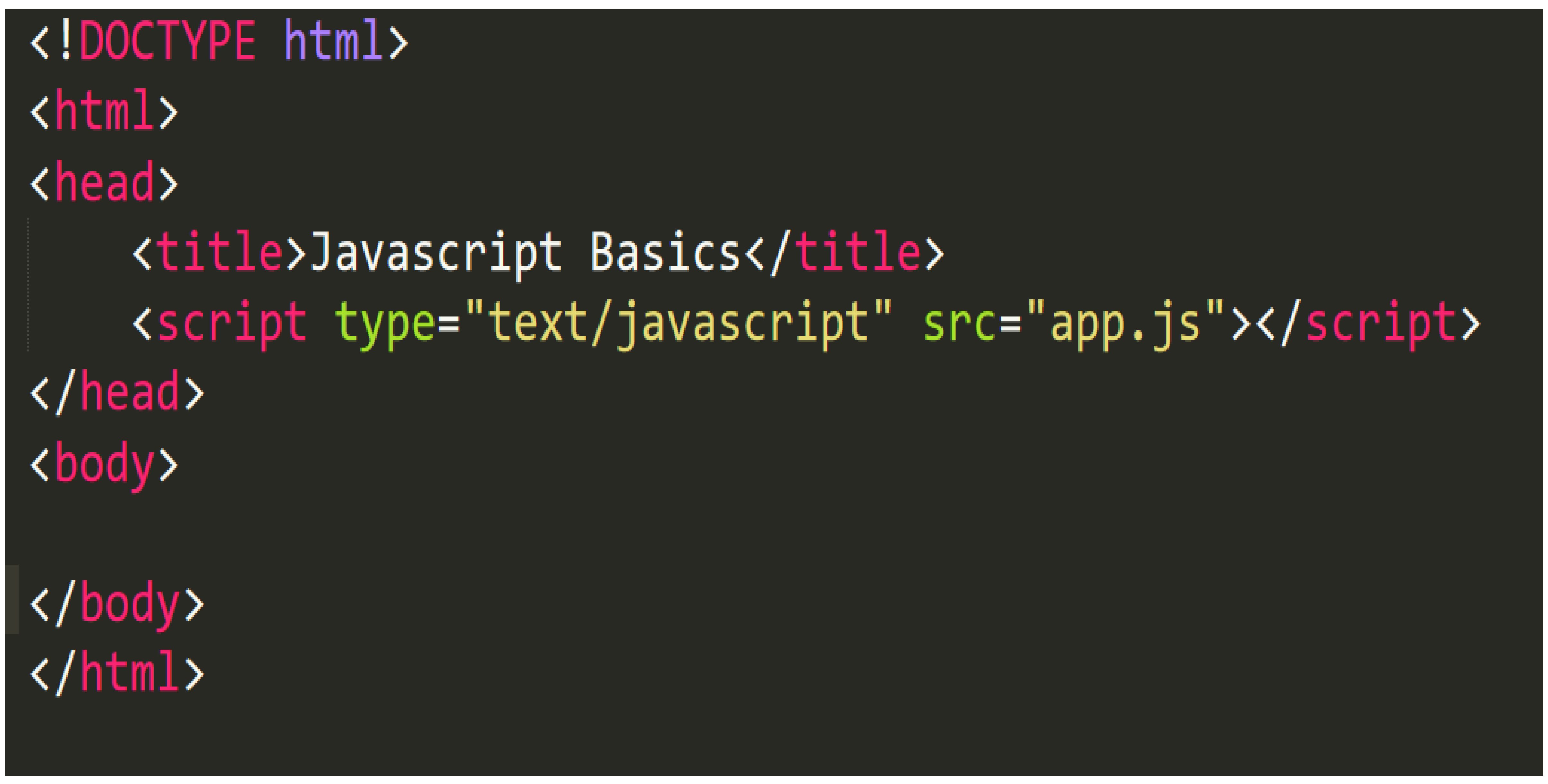
Linking javascript to html file

1) use javascript inside the html file



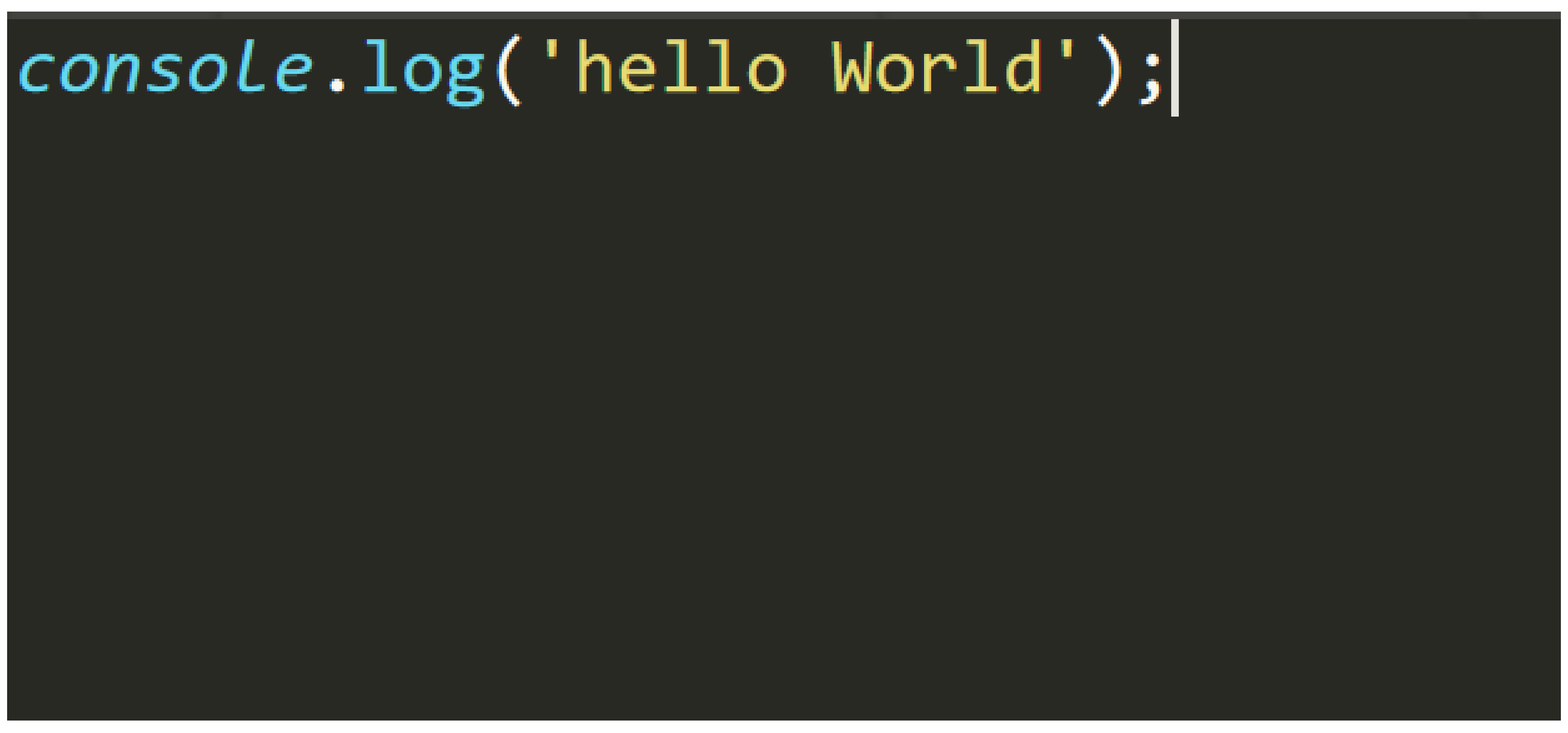
```html
<!DOCTYPE html>
<html>
<head>
    <title>Javascript Basics</title>
</head>
<body>

</body>

 <script type="text/javascript">
    console.log('hello Ro-HiT');
 </script>
</html>
```

2) link to an external javascript file to an html file, in given example we have app.js file link to an our html file now our full javascript code will go into an app.js file



```html
<!DOCTYPE html>
<html>
<head>
    <title>Javascript Basics</title>
    <script type="text/javascript" src="app.js"></script>
</head>
<body>

</body>
</html>
```

app.js file



```javascript
console.log('hello World');
```

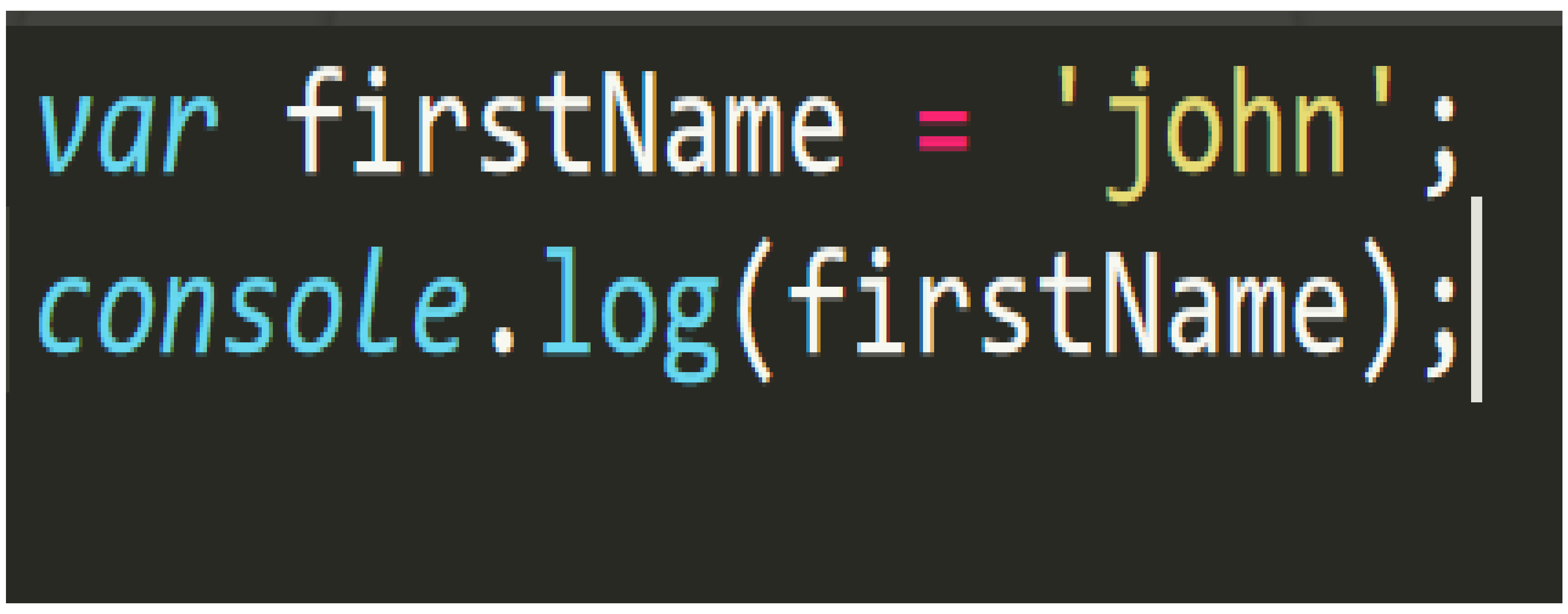
rohit_virkud

Variables & Data types

Variables are the fundamental concept of every programming language in the world but what actually the variable is ?. well

we can say the variable is like a container in which we can store a value in order to use it over and over again in our code,instead of using that value again and again we give that value a name and use that name everywhere in our code

Ex :- var firstName = ‘ john ’;



```
var firstName = 'john';
console.log(firstName);
```

Here we declare our first variable , so here our variable name is firstName and value that assigns to that variable is - john

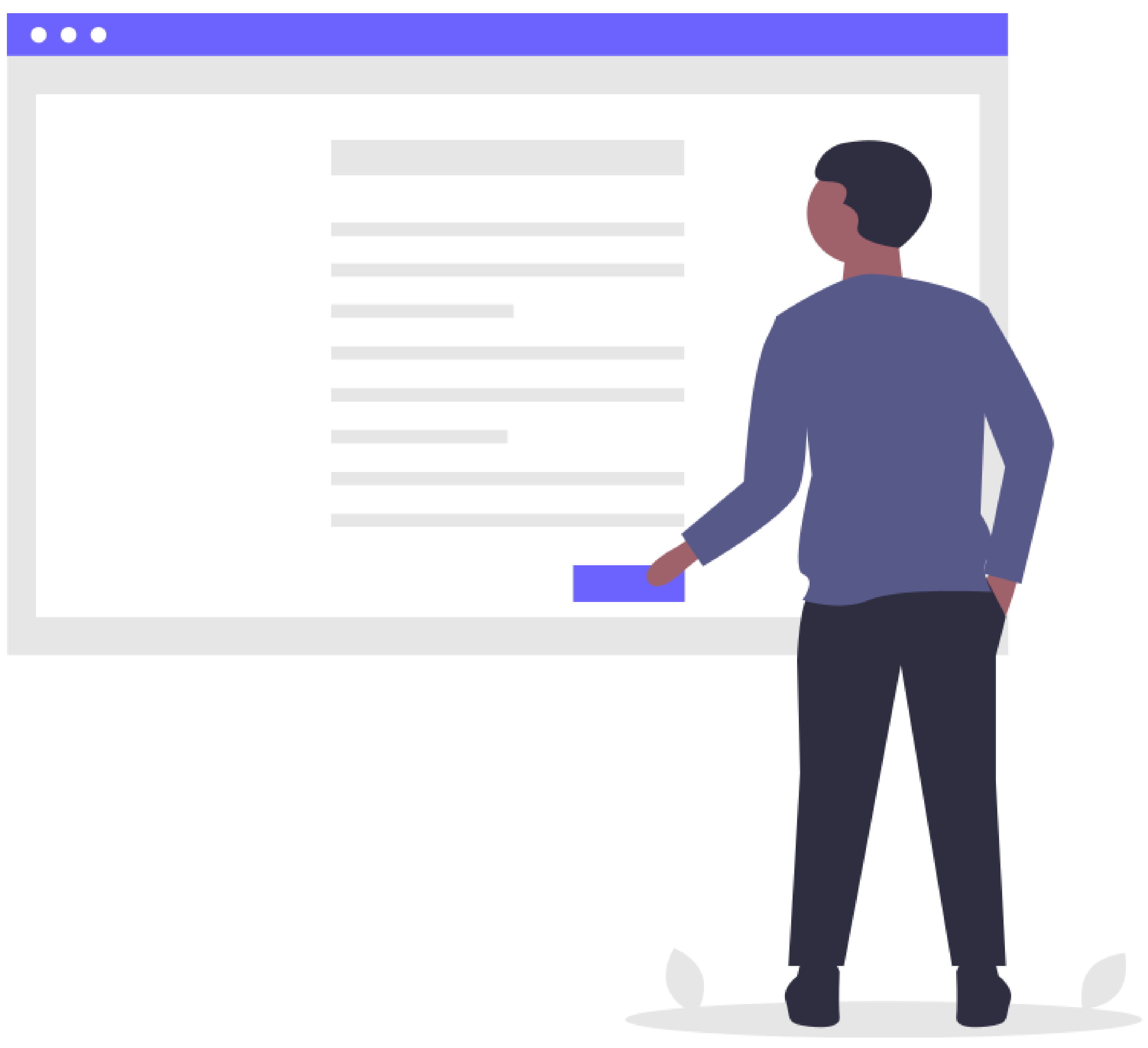
so you can basically think of this firstName variable is a piece of memory into your computer, which stores the value john & it’s a string because it’s written inside the ‘ ’ quotes

to declare text or string we use ‘ quotes ’ in javascript

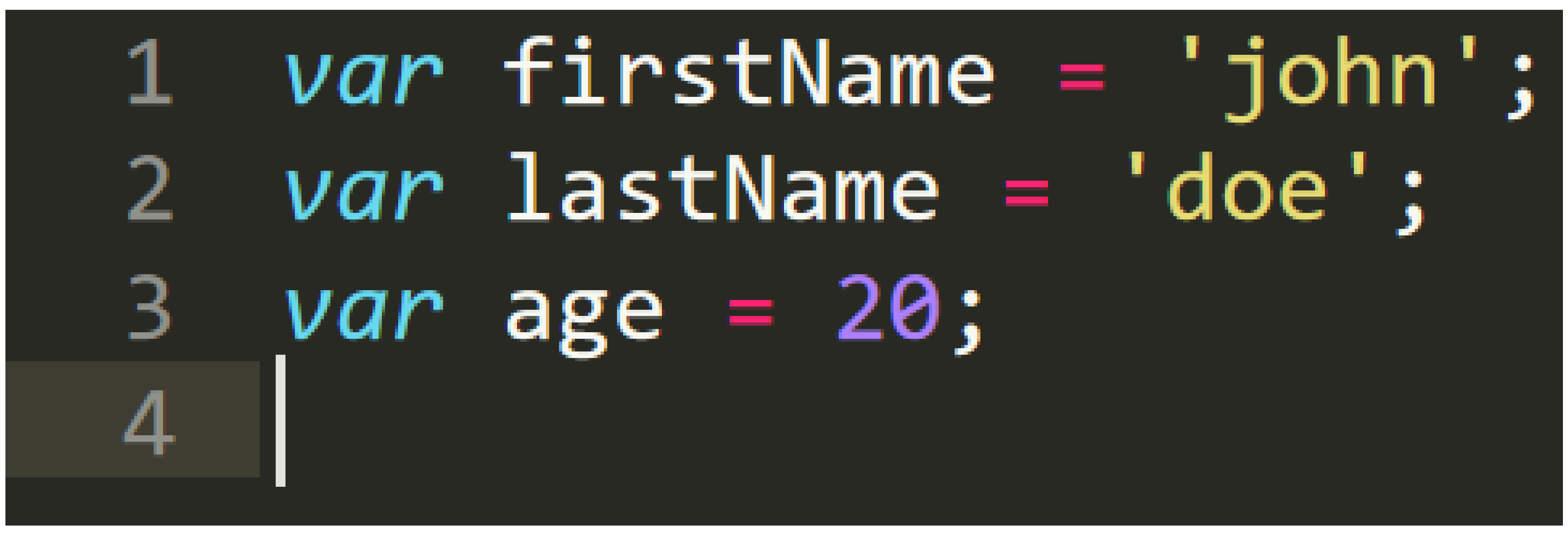
console.log(firstName);

it prints john as a output on our console.

rohit_virkud



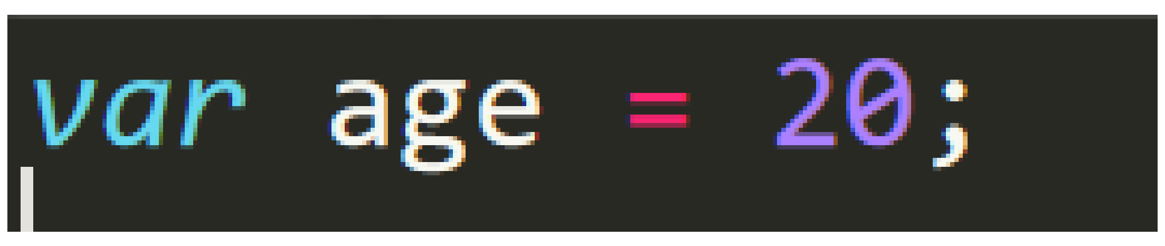
var firstName = ‘john’;
var lastName = ‘doe’;
var age = 20;



```
var firstName = 'john';
var lastName = 'doe';
var age = 20;
```

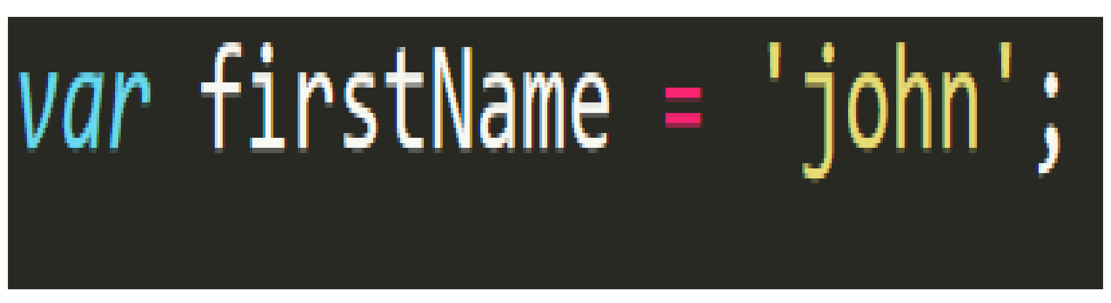
In javascript there are five different datatypes

1) Number :- floating point numbers for decimals & integers every number is a float in javascript even it look like and integer



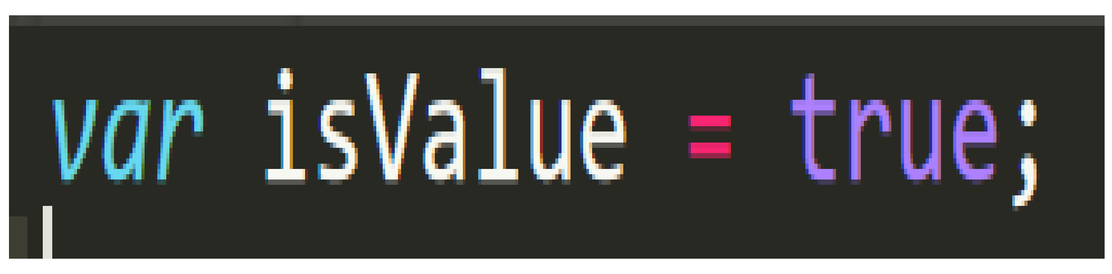
```
var age = 20;
```

2) String :- sequence of characters used for text



```
var firstName = 'john';
```

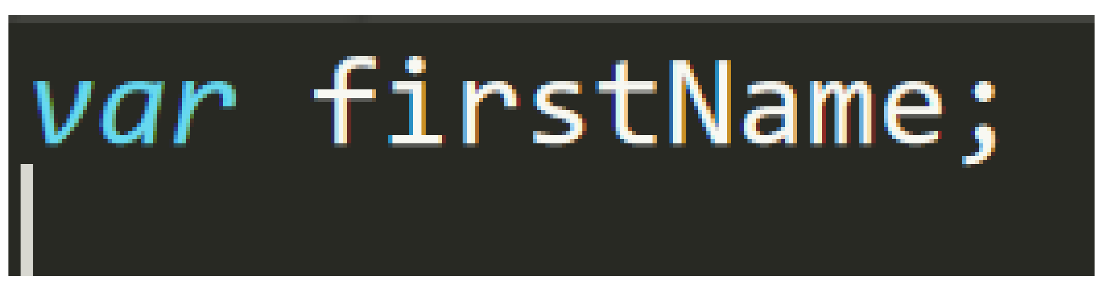
3) Boolean :- logical data types that can be only true and false



```
var isValue = true;
```

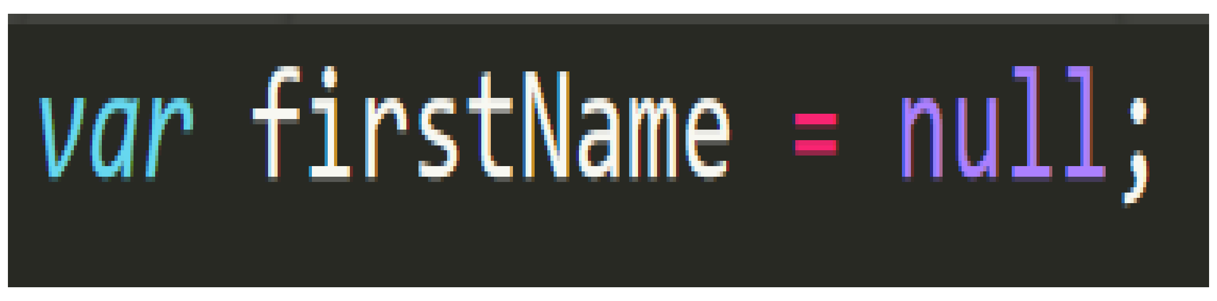
rohit_virkud

4) Undefined :- data type of variable that does not have any value yet

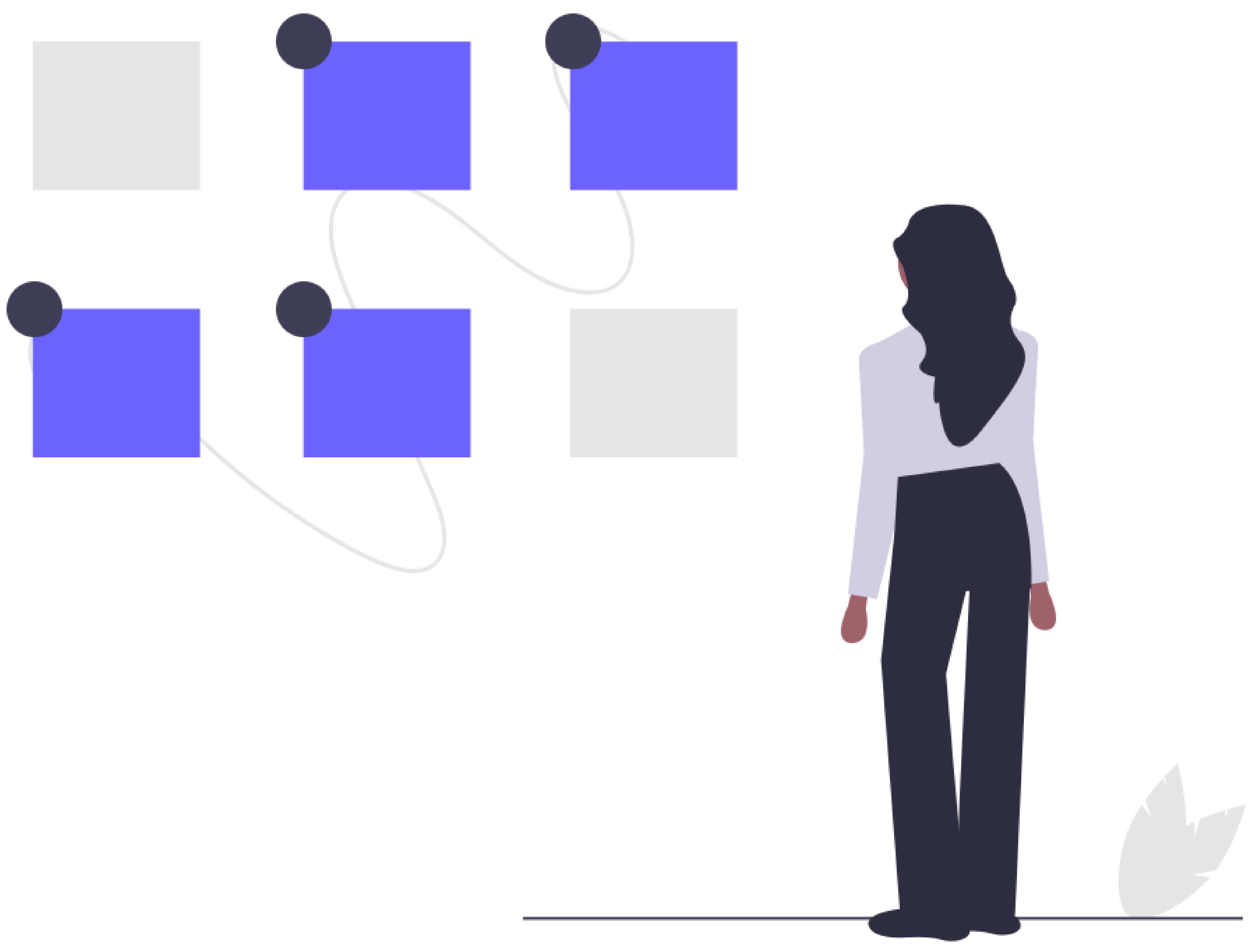


```
var firstName;
```

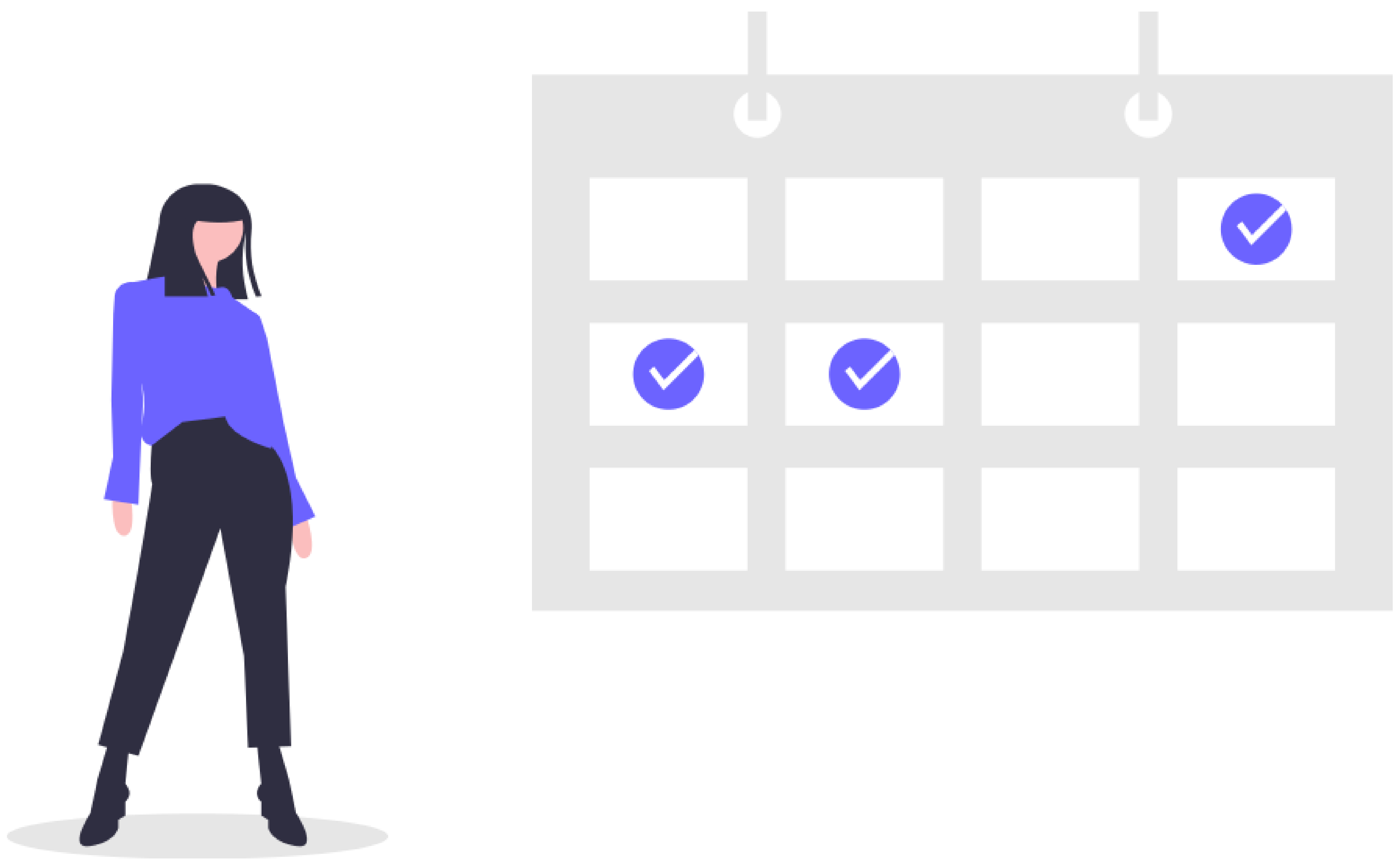
5) Null :- means non-existent



```
var firstName = null;
```



Javascript has a feature called dynamic typing this means we don't have to tell javascript that this variable is a string or number. javascript automatically figures it out that the variable is a string,number or undefined



rohit_virkud

Variable Defination Rules

All you have to know is variable name cannot start with numbers and symbols except with $ signs and underscore

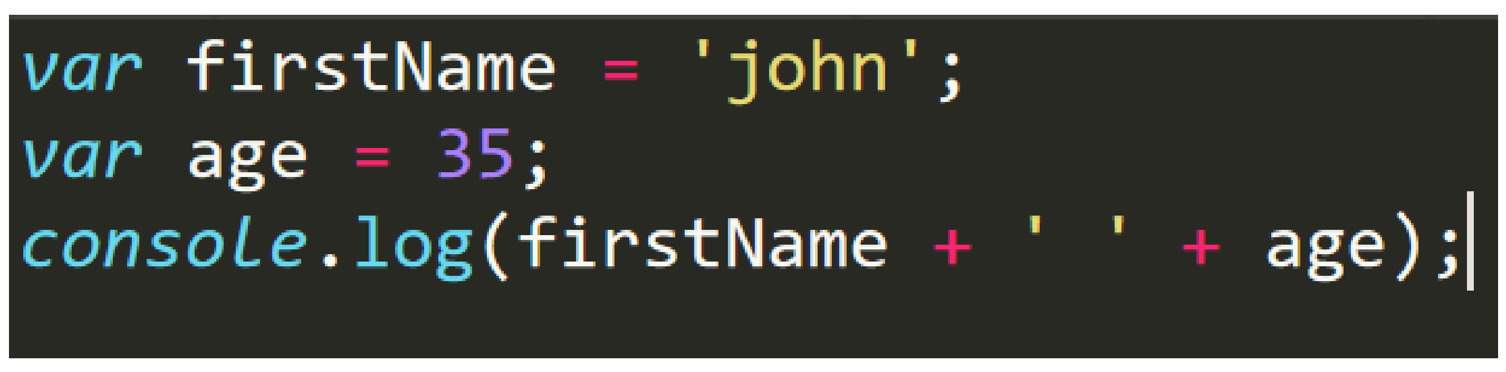
var _firstName - valid	var 3firstName - not valid
var $firstName - valid	var &fistName - not valid

&
we cannot use reserved javascript keywords as variable names just like var,true,false are the names of few

rohit_virkud

Variable Mutation & Type Coercion

Let's learn this from example : -

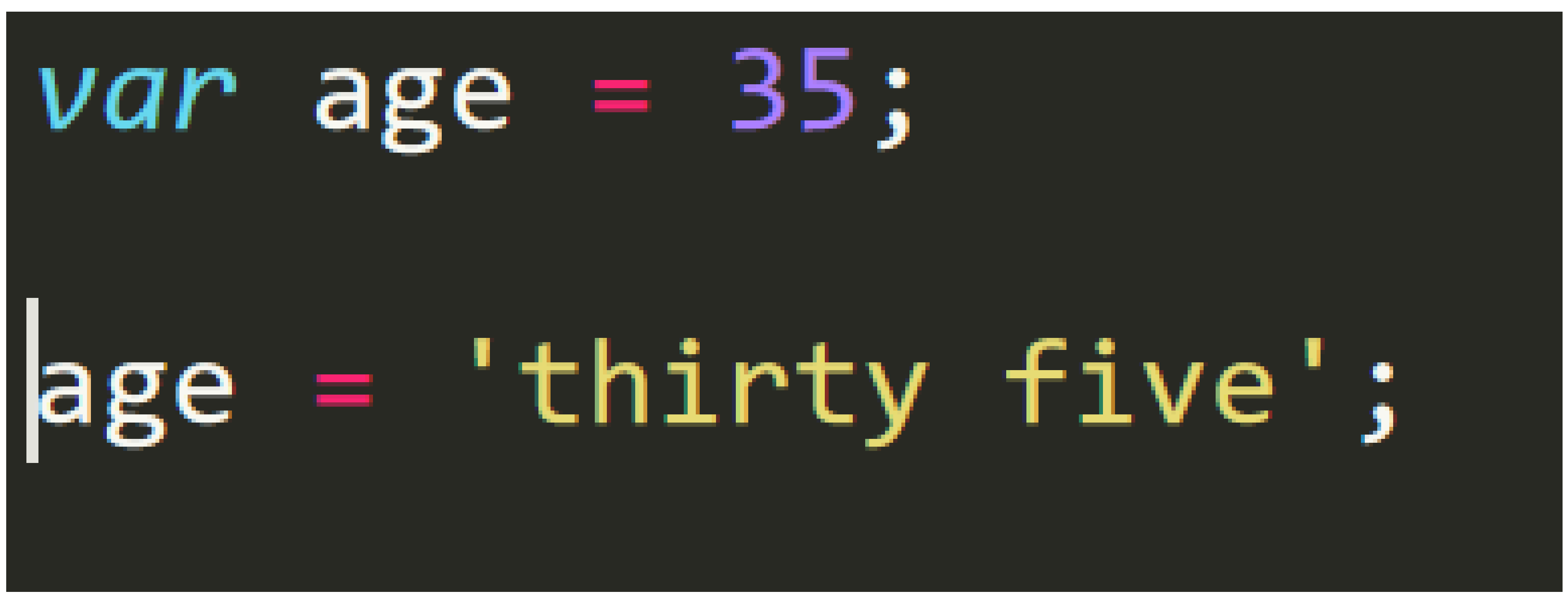


```
var firstName = 'john';
var age = 35;
console.log(firstName + ' ' + age);
```

this works.
thanks to something called type coercion.
So what this means,
javascript automatically converts type from one to another as needed.

You will notice that the variables use different data types, one is a string and the other is a number. Using type coercion, JavaScript will automatically convert types when it is required. In this particular case in order to complete the string it had to convert the number to a string

And Variable Mutation :-



```
var age = 35;

age = 'thirty five';
```

When you mutate a variable you are modifying the original value.
As the variable was previously assigned you do not have to use the var keyword. This time we are are assigning a string value to it, but JavaScript automatically detects the data type and changes it .

rohit_virkud

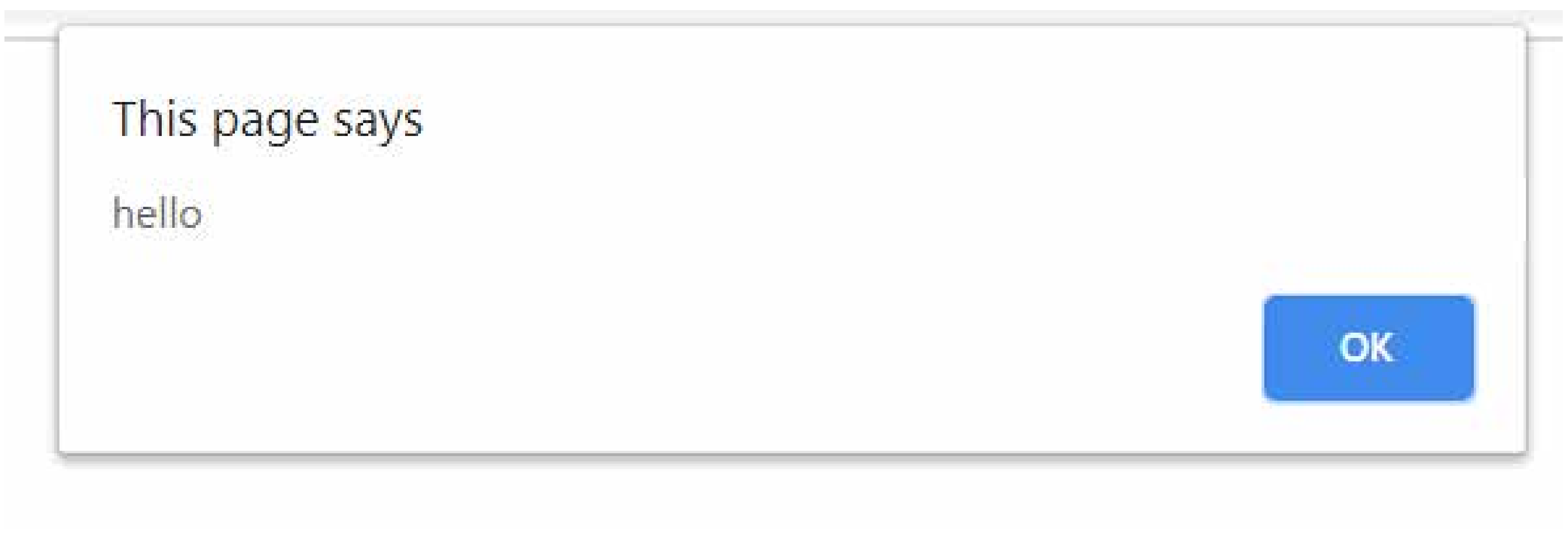
Alert and Prompt in javascript

alert is simply we can say console.logs alternative

```
alert('hello');
```

The alert() method displays an alert box with a specified message and an OK button.

An alert box is often used if you want to make sure information comes through to the user.

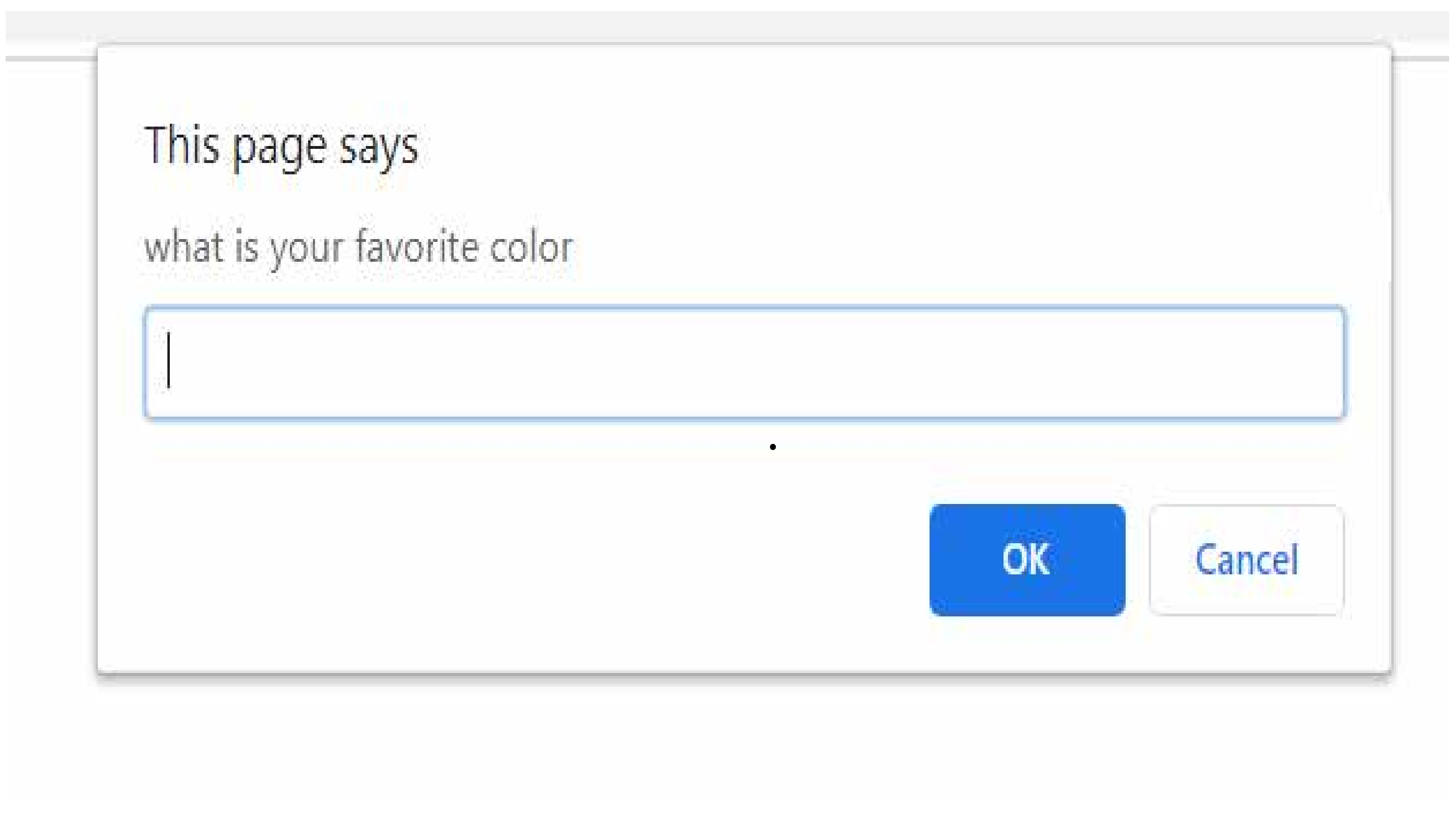

This page says

hello

```
prompt('what is your favorite color');
```

The prompt() method displays a dialog box that prompts the visitor for input.

A prompt box is often used if you want the user to input a value before entering a page


This page says

what is your favorite color

rohit_virkud

BASIC OPERATORS (+ , -)

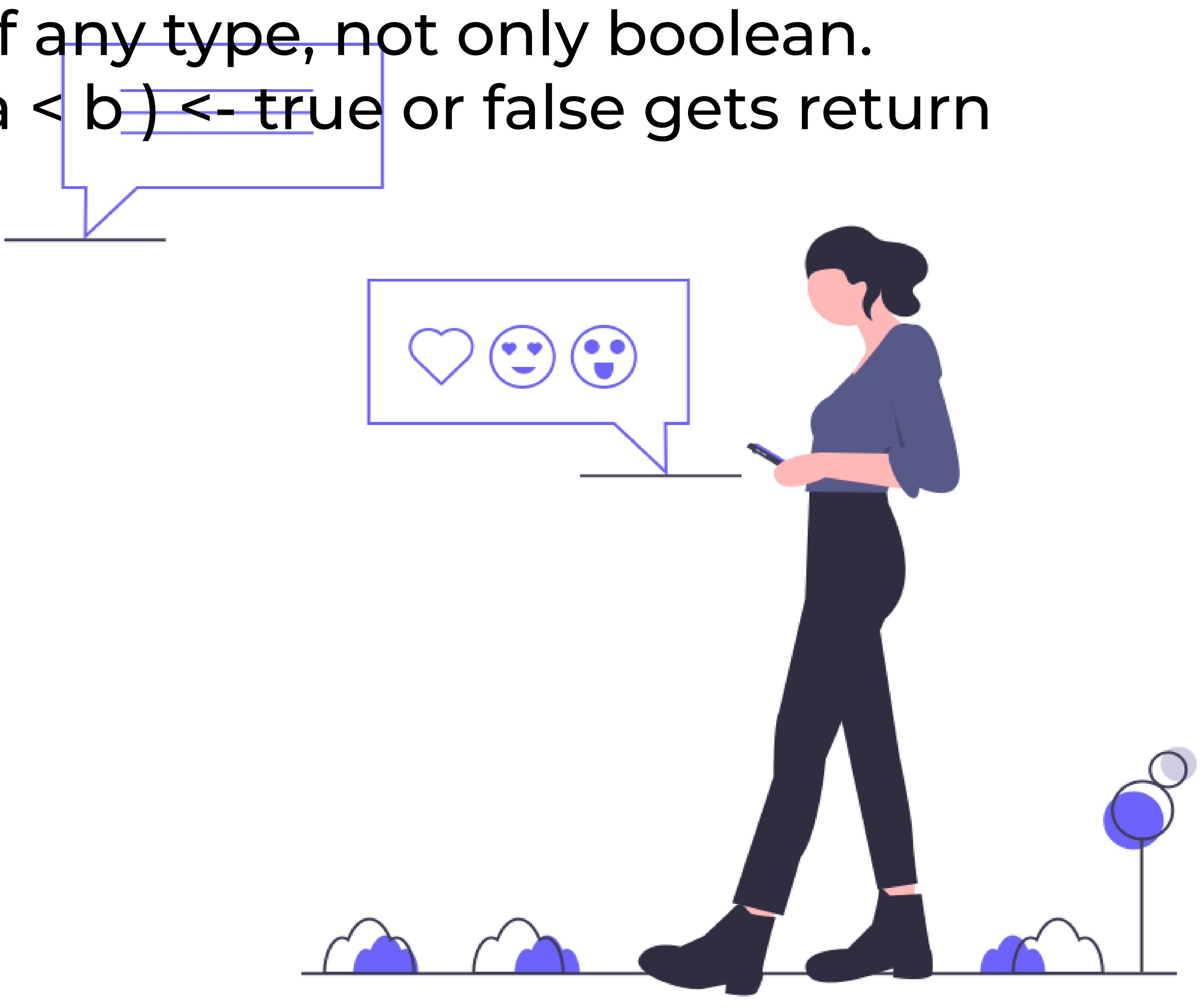
* Operators are simply function which written in special way in javascript

Let's calculate the john's born year :-
var johnCurrentAge = 20;
var johnYear = 2020 - johnCurrentAge;

LOGICAL OPERATORS

There are three logical operators in JavaScript: || (OR), && (AND), ! (NOT).

Although they are called "logical", they can be applied to values of any type, not only boolean.
ex :- if(a < b) <- true or false gets return



Basic Boolean Logic

AND	TRUE	FALSE
TRUE	TRUE	FALSE
FALSE	FALSE	FALSE

OR	TRUE	FALSE
TRUE	TRUE	TRUE
FALSE	TRUE	FALSE

NOT (!) :- if a = true !a = false,
if a = false !a = true

OPERATOR PRECEDENCE

Which operator will proceed first is operator precedence

reference :-
https://en.cppreference.com/w/c/language/operator_precedence

More Operators :-

```
var  x = 5;
x = x * 2; // we can write this in more simple way like
                  x *= 2
x = x+2;  // x += 2
```

if we have to add only one then we can use

```
x++ ;
```

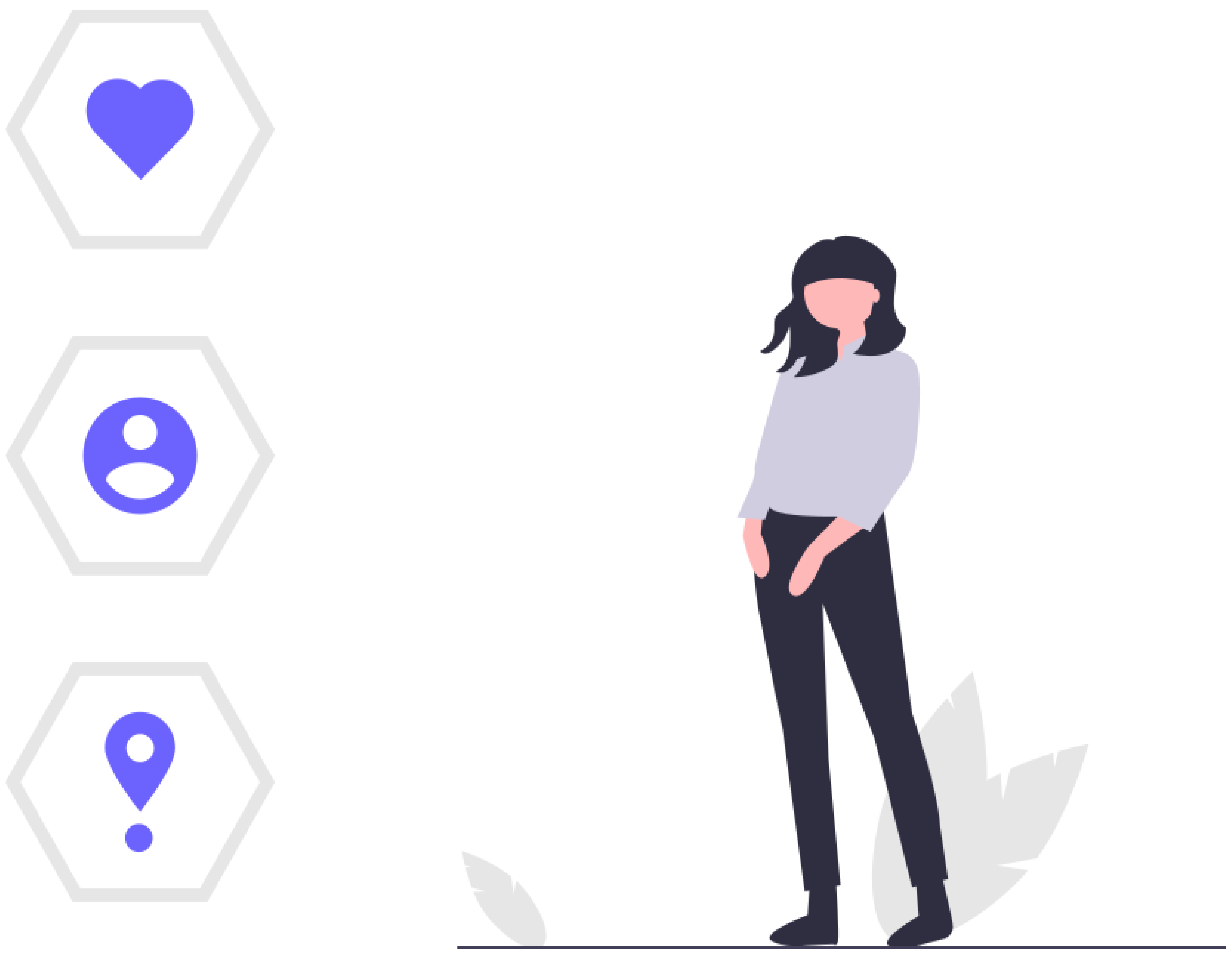
rohit_virkud

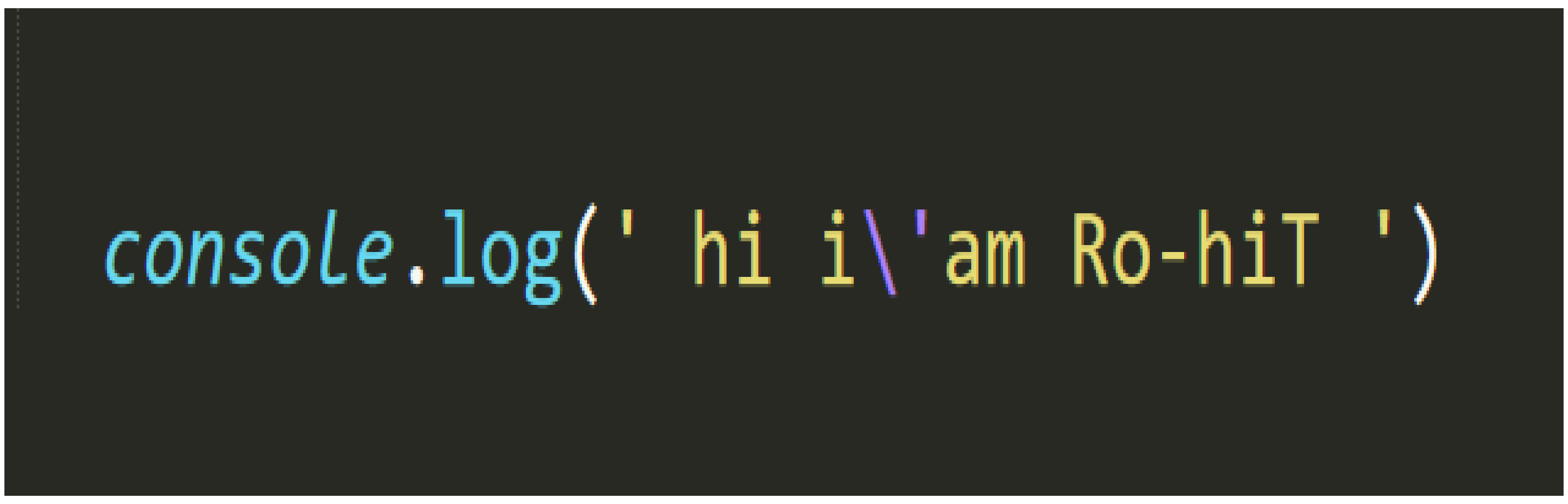
Escaping with \

console.log(‘ hi i’am Ro-hiT '); ✗

Here it gives us an error
’ hi i ' it gets closed
to manage this error we have to
ignore that middle quote **' for that**
we use \ symbol

console.log(‘ hi i\’am Ro-hiT '); ✓





```
console.log(' hi i\'am Ro-hiT ')
```

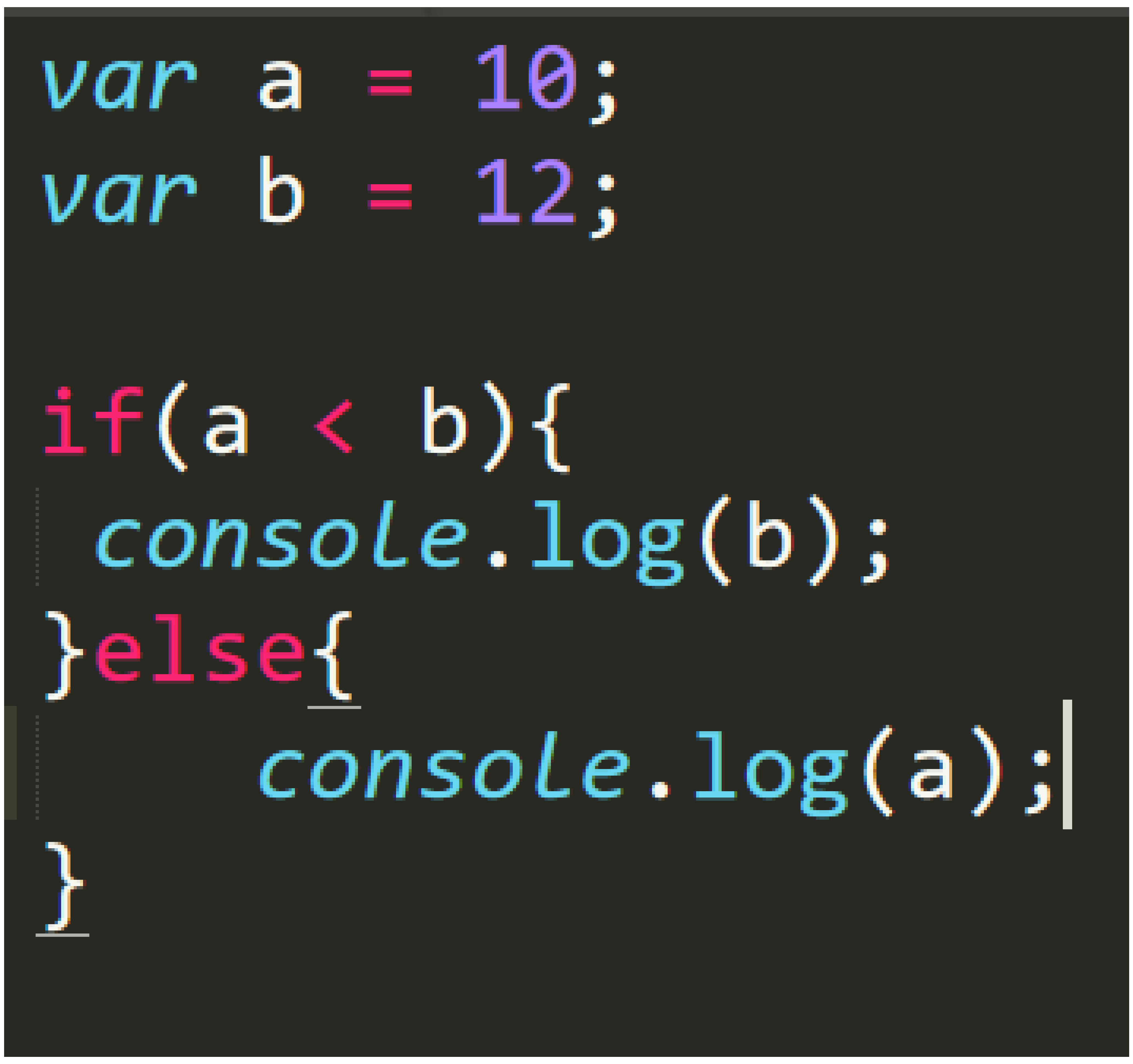
rohit_virkud

CONTROL STRUCTURE OR STATEMENT

“control structures” are the structures used to control the flow of a program.

if - else

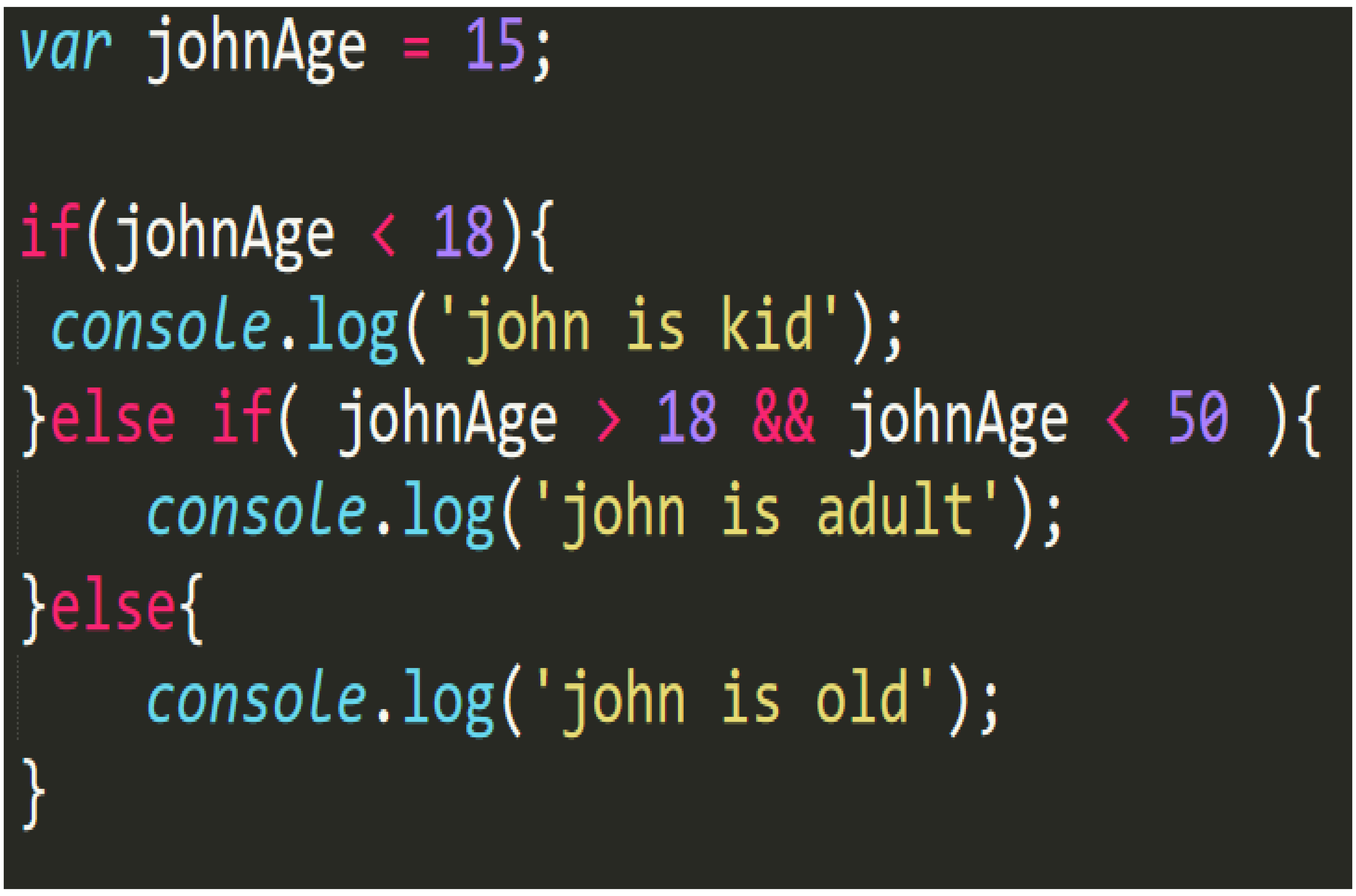
```
if ( condition ) {

    [redacted]

} else {
    [redacted]
}
```



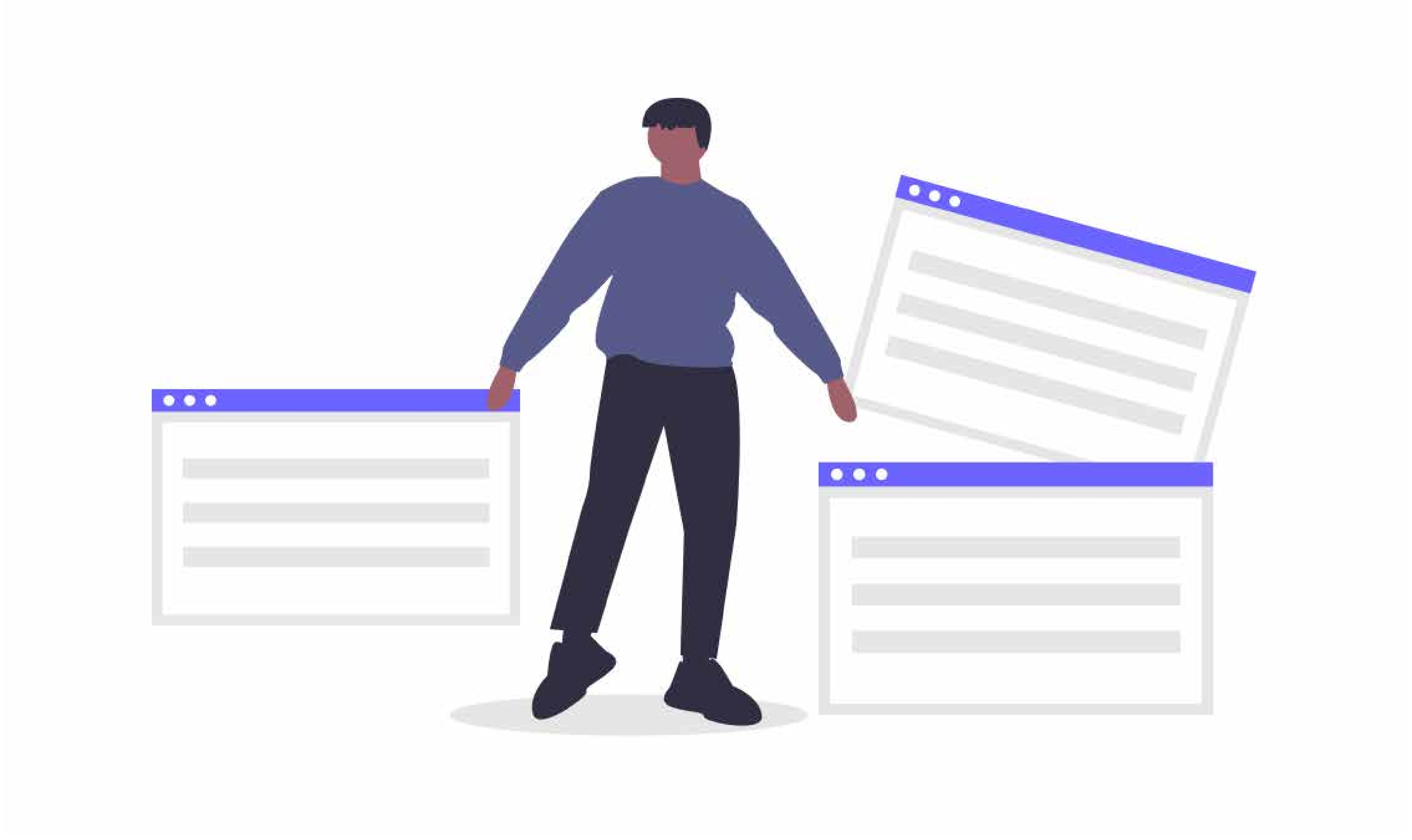
```
var a = 10;
var b = 12;

if(a < b){
 console.log(b);
}else{
     console.log(a);
}
```

rohit_virkud

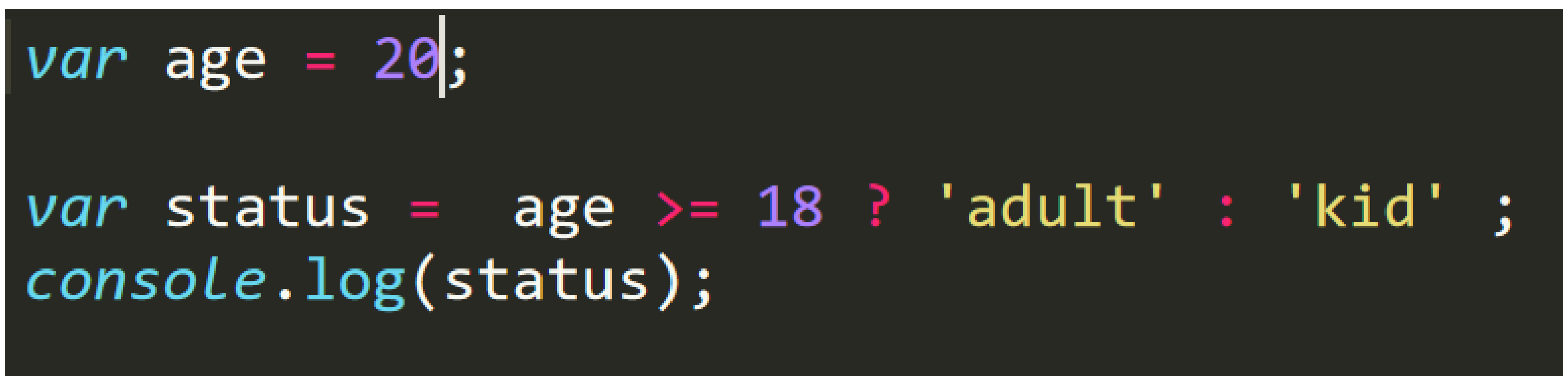
if - elseif - else



```
var johnAge = 15;

if(johnAge < 18){
 console.log('john is kid');
}else if( johnAge > 18 && johnAge < 50 ){
    console.log('john is adult');
}else{
    console.log('john is old');
}
```



TERNARY OPERATOR



```
var age = 20;

var status =  age >= 18 ? 'adult' : 'kid' ;
console.log(status);
```

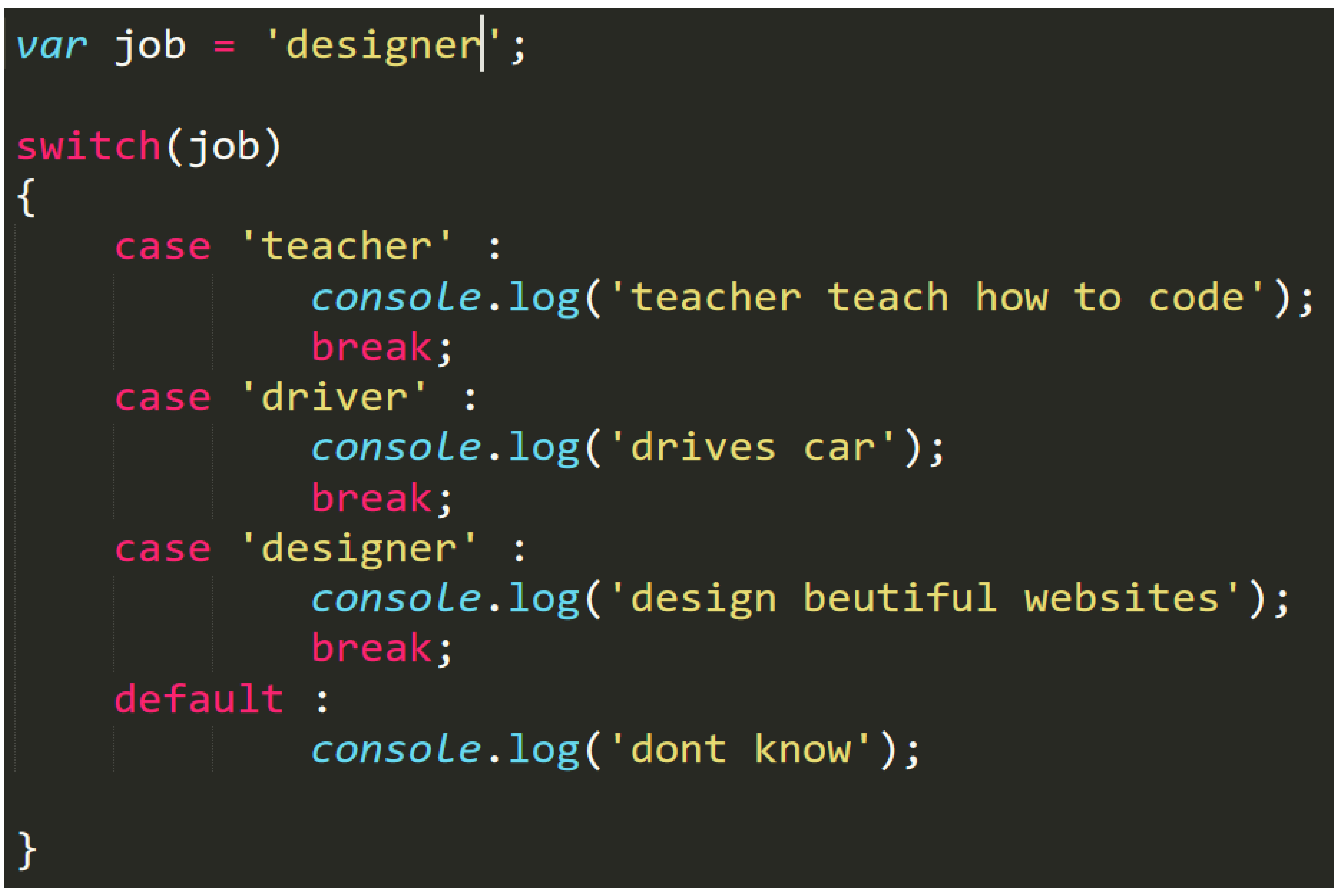
if else in a simple way is a ternary operator using only three operators so basically the structure here is quite simple :: condition ? result : else result

if(condtion) ? if block execution : else block execution

rohit_virkud

SWITCH STATEMENT

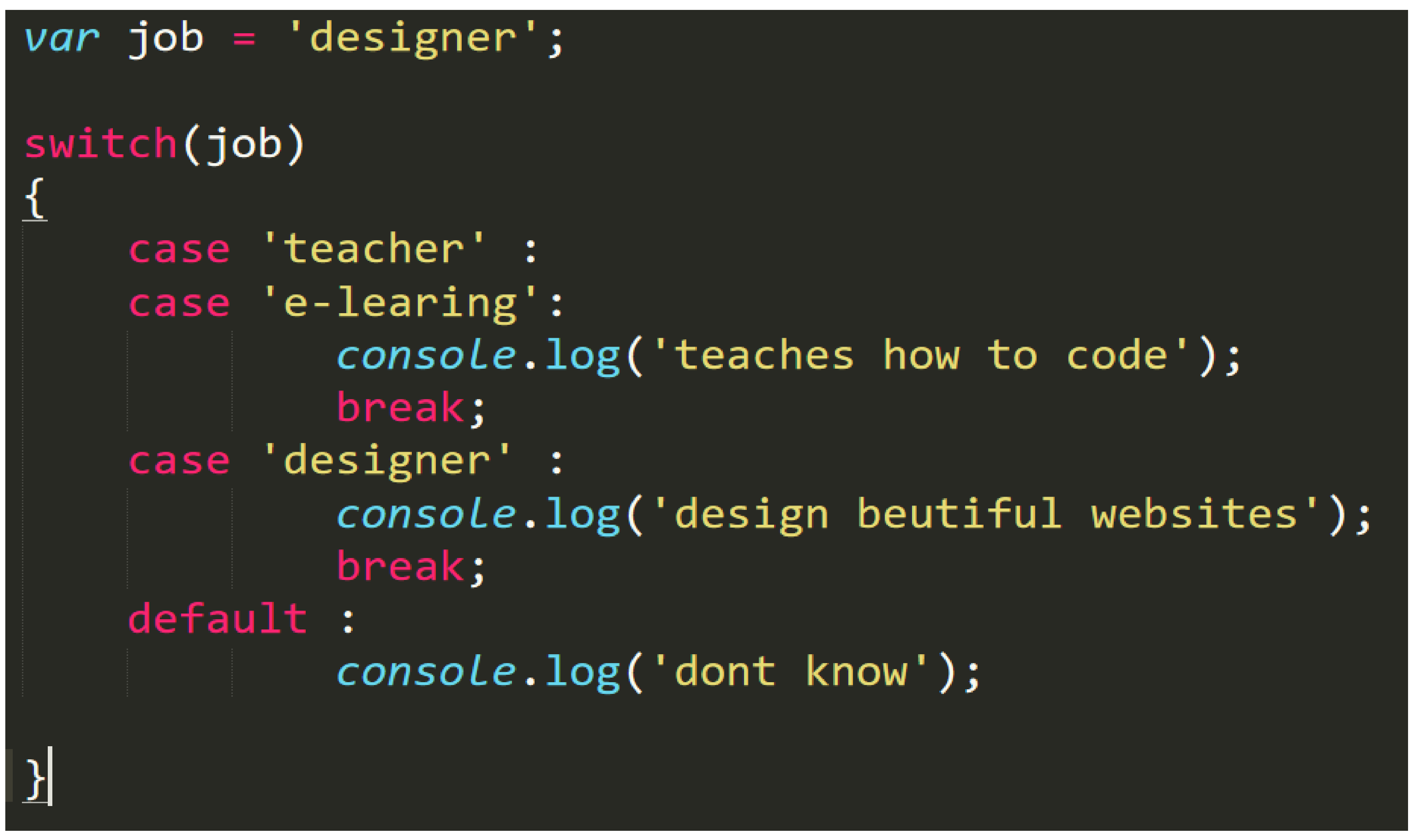
switch is a multiple if-else



```javascript
var job = 'designer';

switch(job)
{
    case 'teacher' :
            console.log('teacher teach how to code');
            break;
    case 'driver' :
            console.log('drives car');
            break;
    case 'designer' :
            console.log('design beutiful websites');
            break;
    default :
            console.log('dont know');

}
```

```
if ()
{
 .......
}else if()
{
......
}else if()
{
.....
}else{
.........
}
```

```
switch ()
{
 case ‘ first’ :
         .......
         break;
 case ‘ second ’ :
          .........
          break;
 case ‘third’ :
          .......
          break;
  default :
         ........
}
```



```javascript
var job = 'designer';

switch(job)
{
    case 'teacher' :
    case 'e-learing':
            console.log('teaches how to code');
            break;
    case 'designer' :
            console.log('design beutiful websites');
            break;
    default :
            console.log('dont know');

}
```

we can also put two cases at the same time

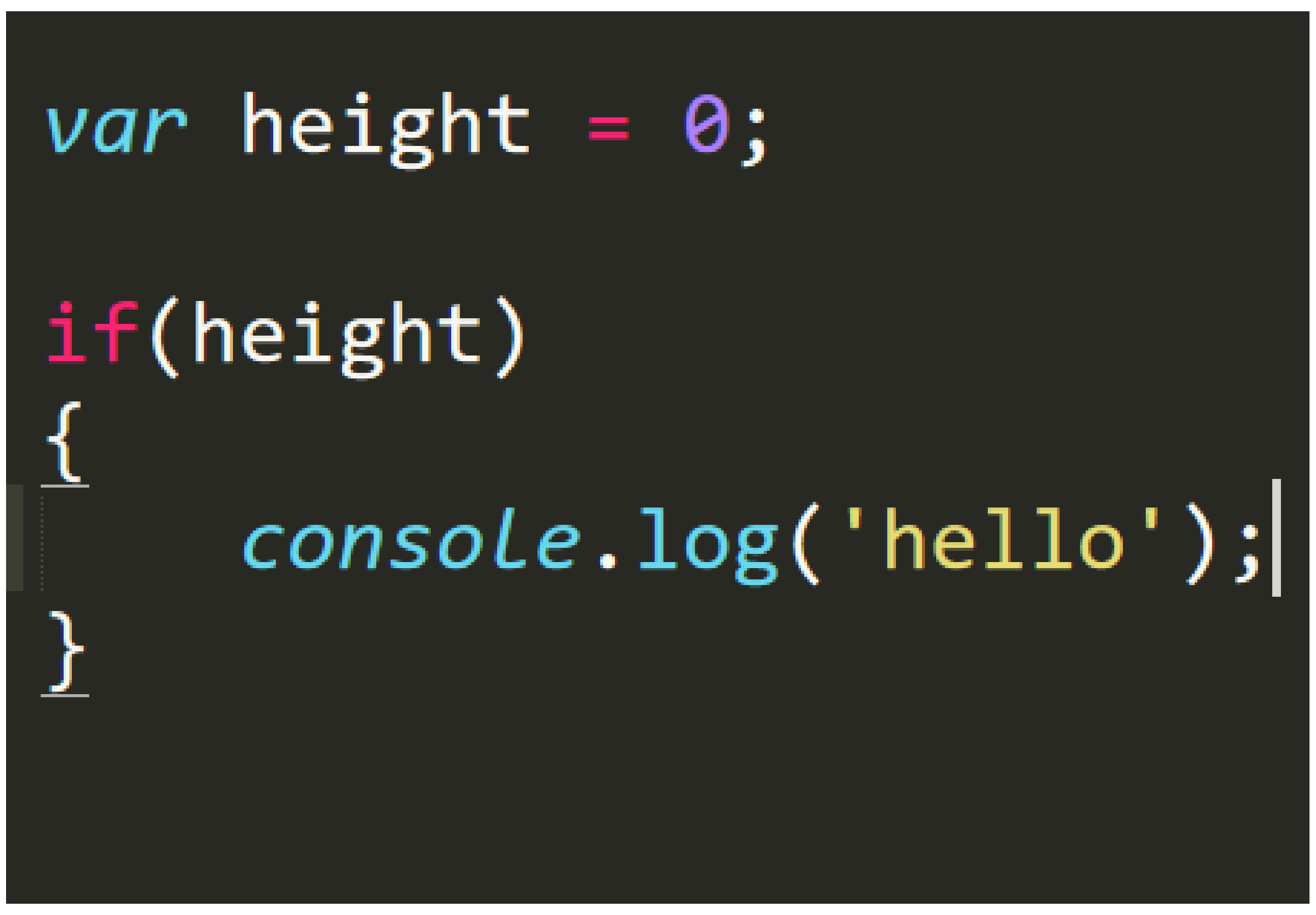
rohit_virkud

Falsy & Truthy Values in JavaScript

undefined, null , 0 , ‘ ’ , NaN
falsy values turns out to be false when evaluated if condition

the values those are not falsy are truthy
defined , 1

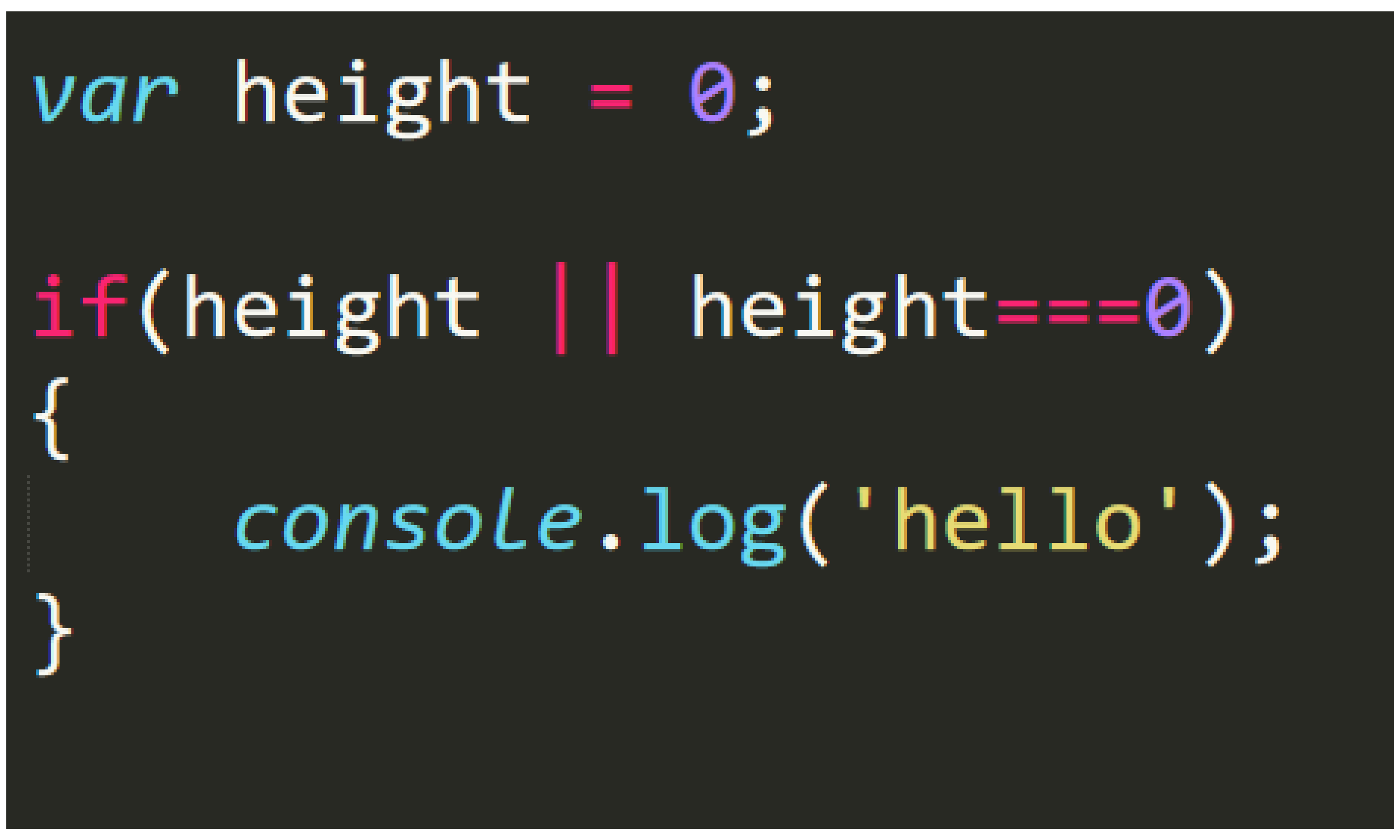
Error



```
var height = 0;

if(height)
{
    console.log('hello');
}
```

Error because 0 is a falsy value

SOLVED



```
var height = 0;

if(height || height===0)
{
    console.log('hello');
}
```

Check height === 0 then it will work fine

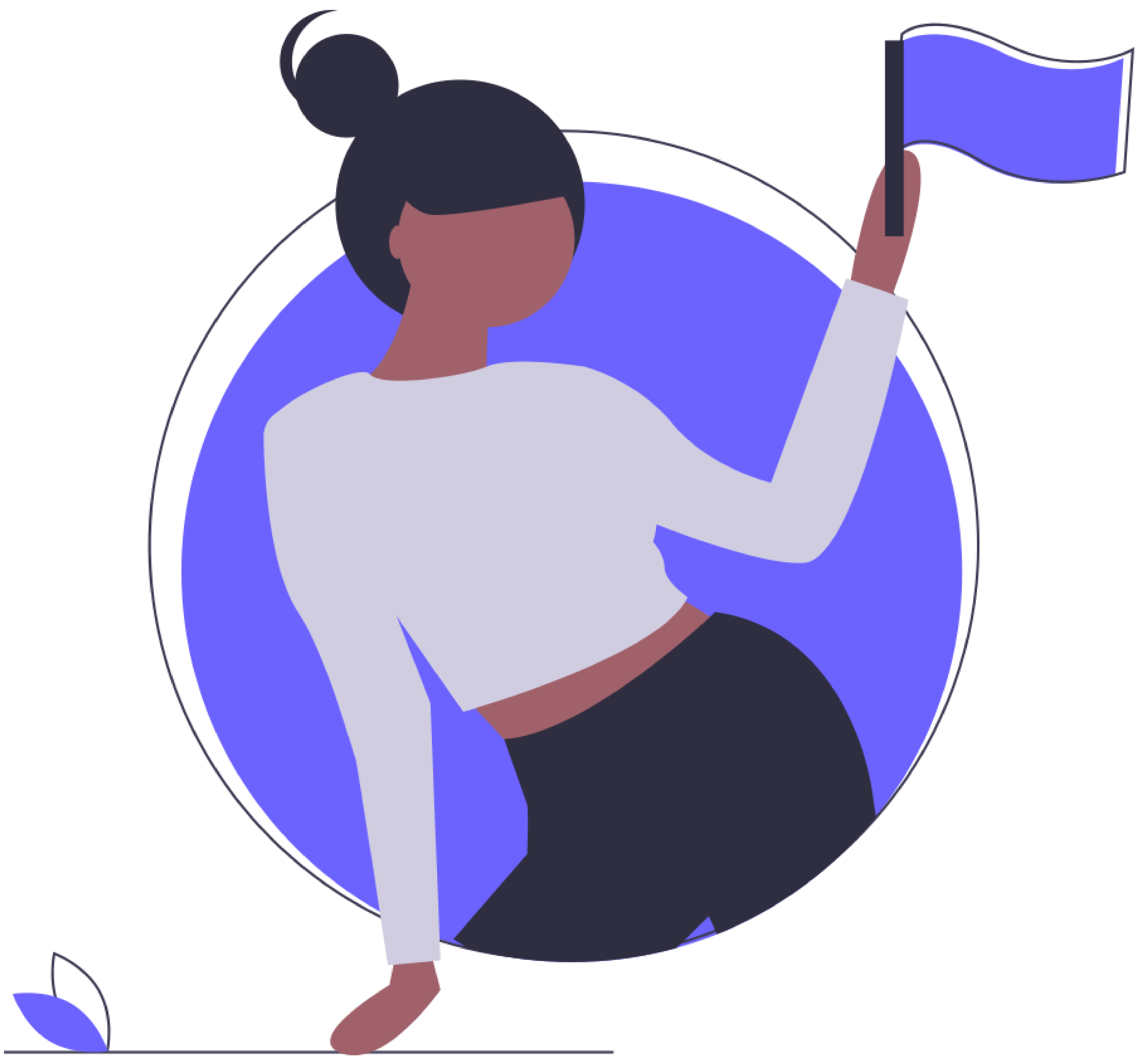
rohit_virkud

EQUALITY OPERATORS

The single = is used for assigning the value to variable
var a = 12;
and == , === are used for comparison purposes.

== compares two variables irrespective of data type (a == b)

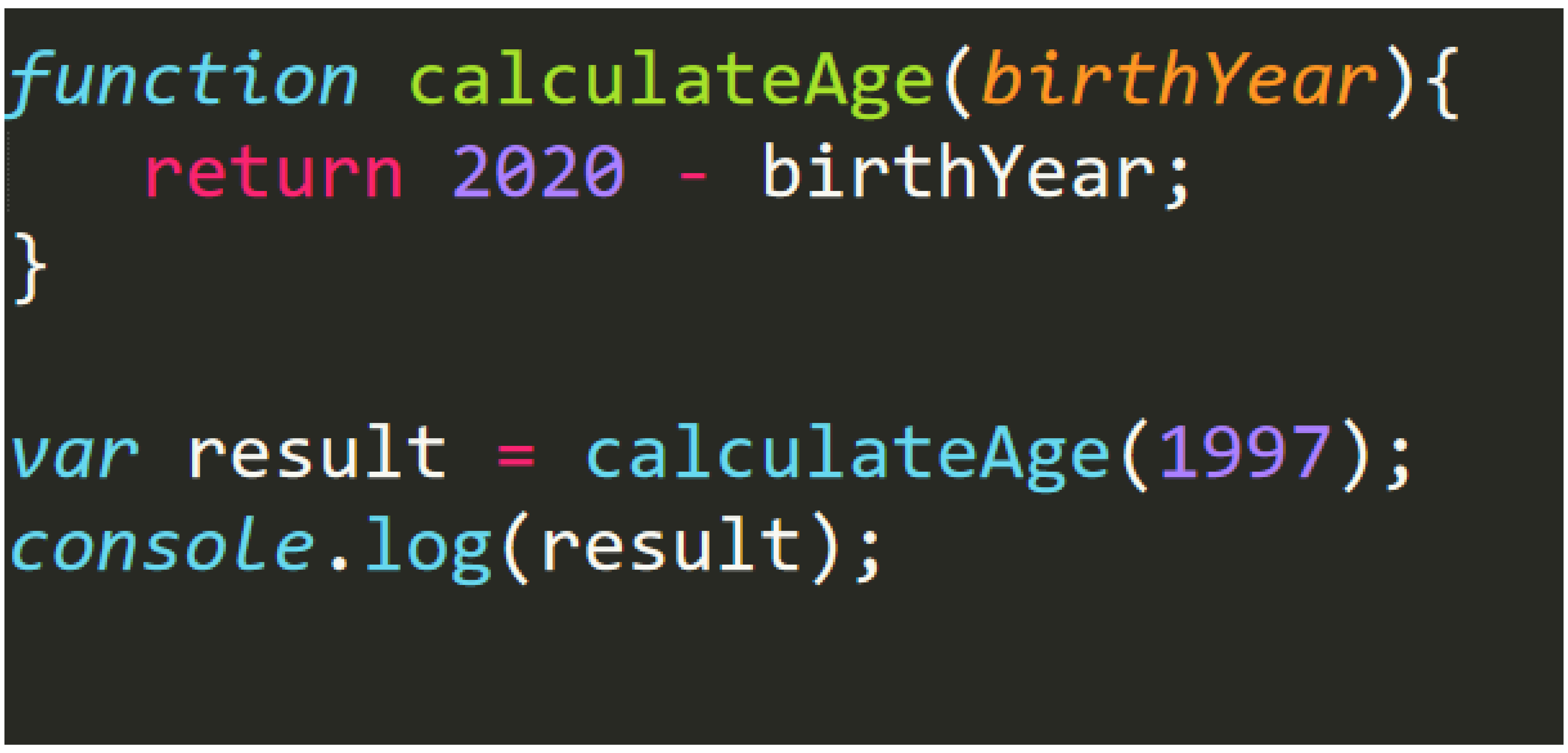
while === compares two variables in a strict check, which means it checks for data type also then it returns true or false (a === b)



rohit_virkud

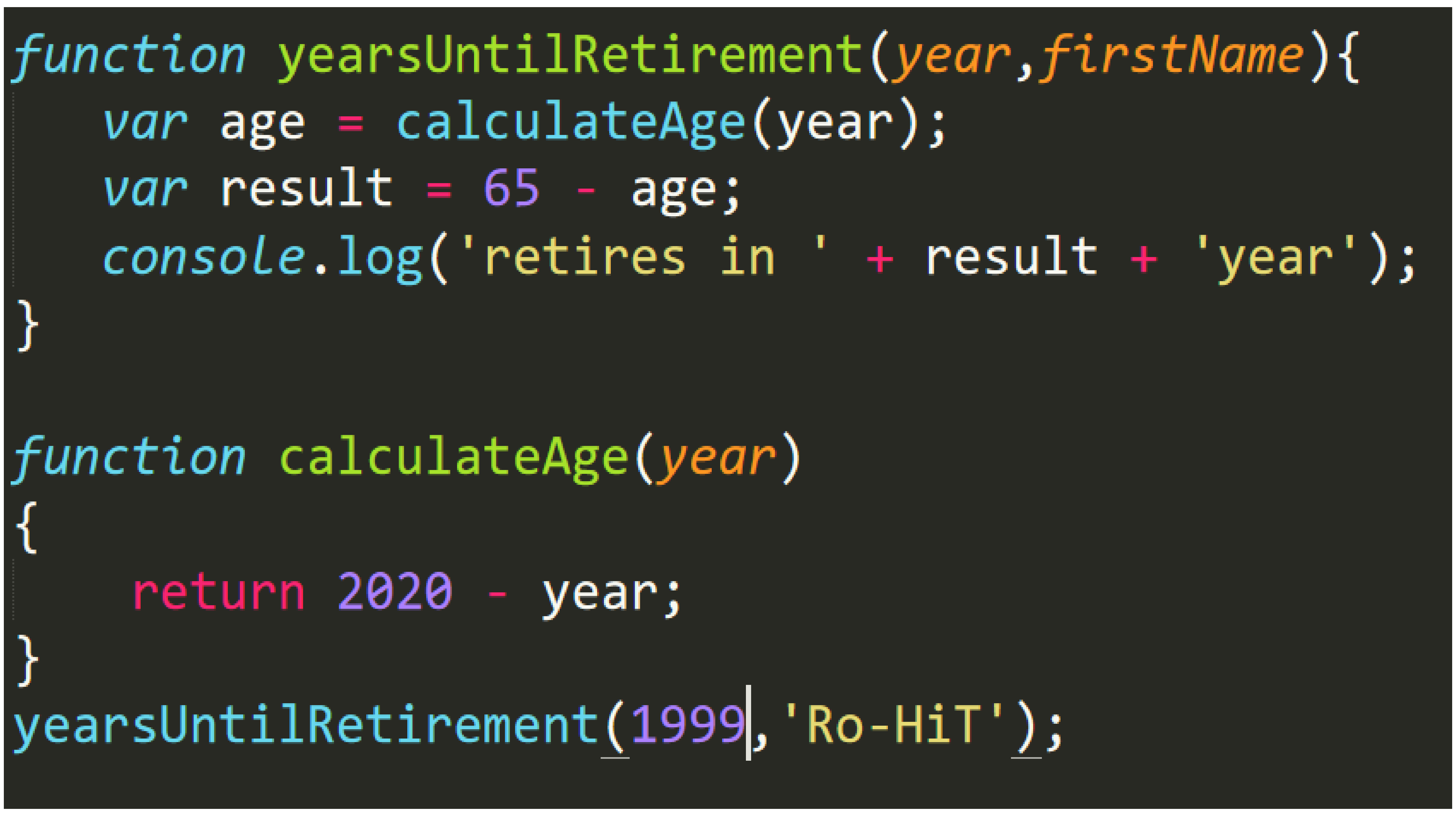
FUNCTIONS

if A is a piece of code we want to run many times, then we use functions instead of writing over and over it again



```
function calculateAge(birthYear){
   return 2020 - birthYear;
}

var result = calculateAge(1997);
console.log(result);
```

we can use function inside function as well



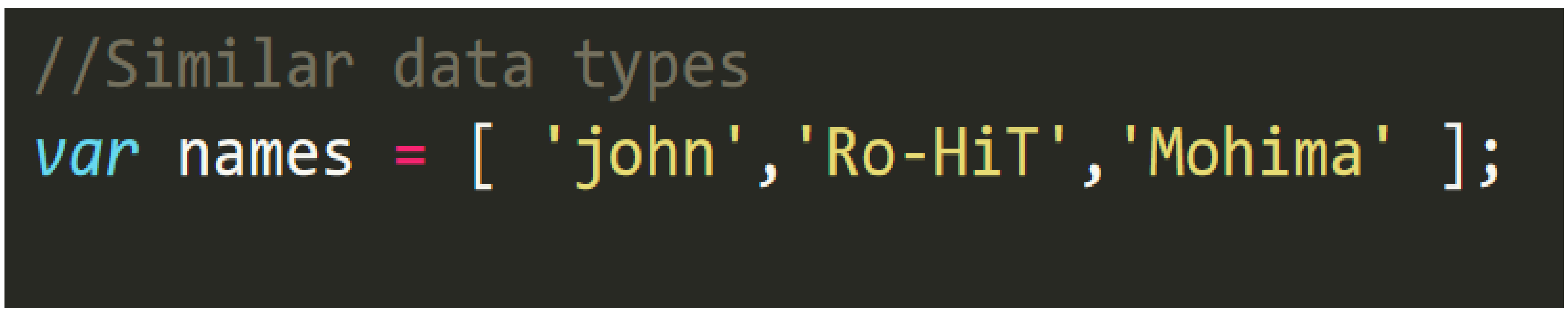
```
function yearsUntilRetirement(year,firstName){
   var age = calculateAge(year);
   var result = 65 - age;
   console.log('retires in ' + result + 'year');
}

function calculateAge(year)
{
    return 2020 - year;
}
yearsUntilRetirement(1999,'Ro-HiT');
```

Return keyword not only returns a value it also closes the function.

rohit_virkud

ARRAYS

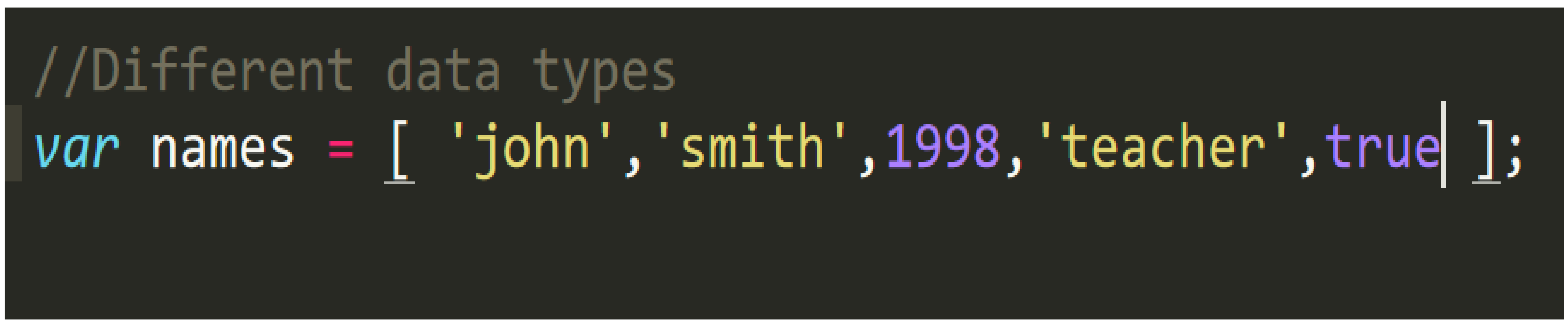
Fundamental concept that you are going use in your entire coding life



```
//Similar data types
var names = [ 'john','Ro-HiT','Mohima' ];
```

Similar data type array

Different data type array



```
//Different data types
var names = [ 'john','smith',1998,'teacher',true ];
```

names.length <- gives you length of an array

names[4] <- gives you fifth element value in the array and also array index starts with 0 so by our 2nd example names[4] means true

we can also change values by
names[4] = ‘ Hero ’;
now names array will be
names = [‘ john ’ , ‘ smith ’ , 1998,‘ teacher ’, ‘ Hero ’]

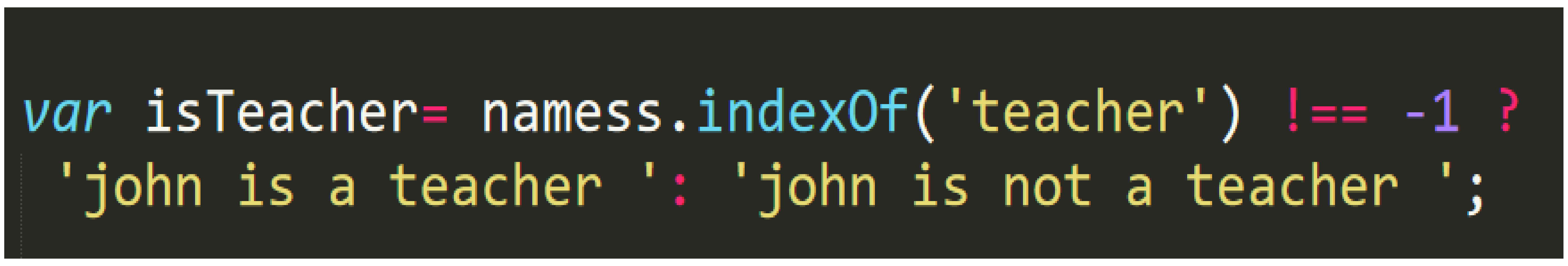
some more :--
names.push(‘ blue ’) <- add blue at the end of the array
names.unshift(‘ Mr ’) <- add Mr at the first in our array

names.pop() <- remove last element
names.shift() <- remove first element

rohit_virkud

names.indexOf(1998) <- shows where 1998 in array returns index , if element is not found it returns -1

we can use that -1 to see that data is present in array or not

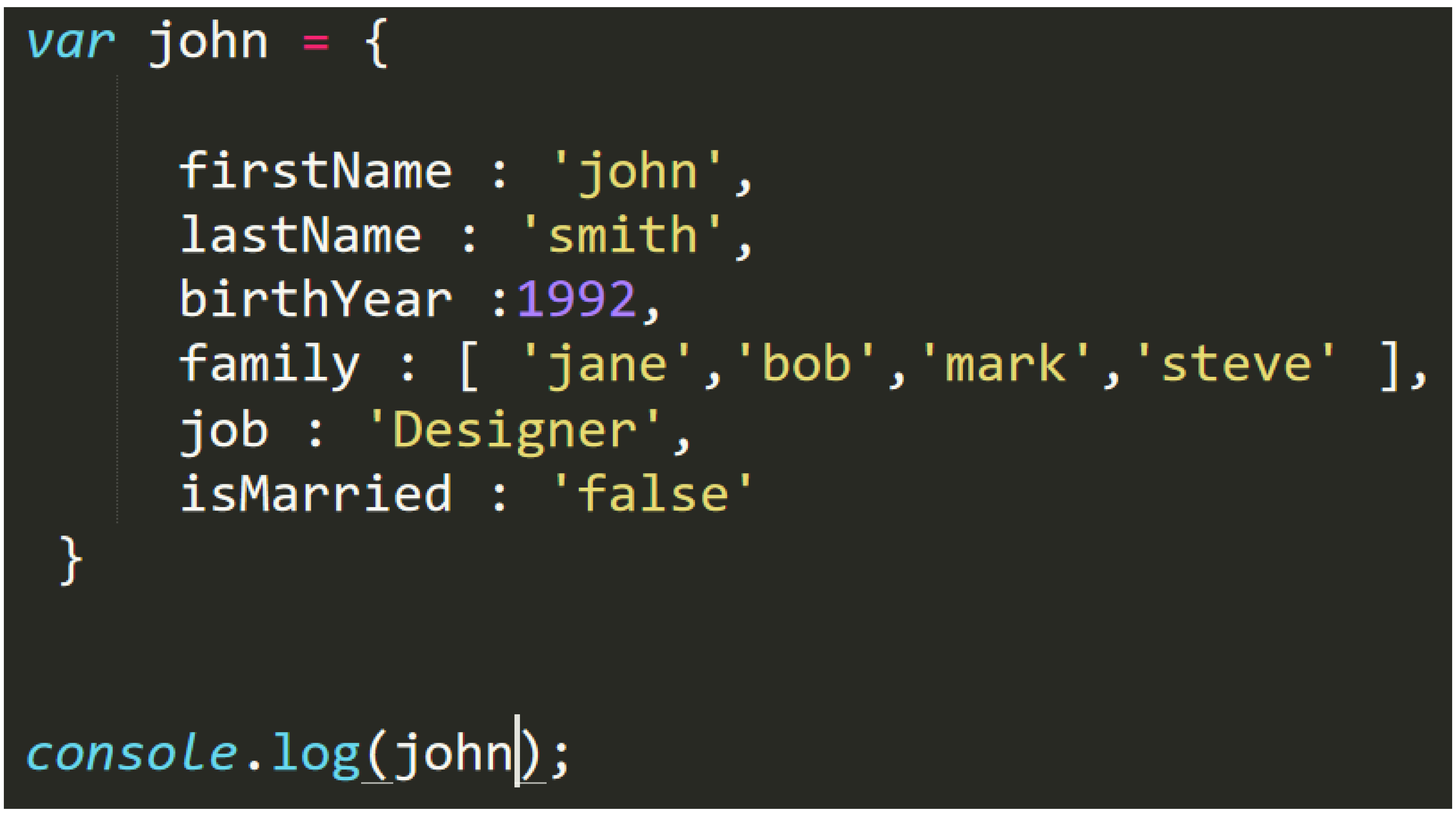


```
var isTeacher= namess.indexOf('teacher') !== -1 ?
 'john is a teacher ': 'john is not a teacher ';
```

rohit_virkud

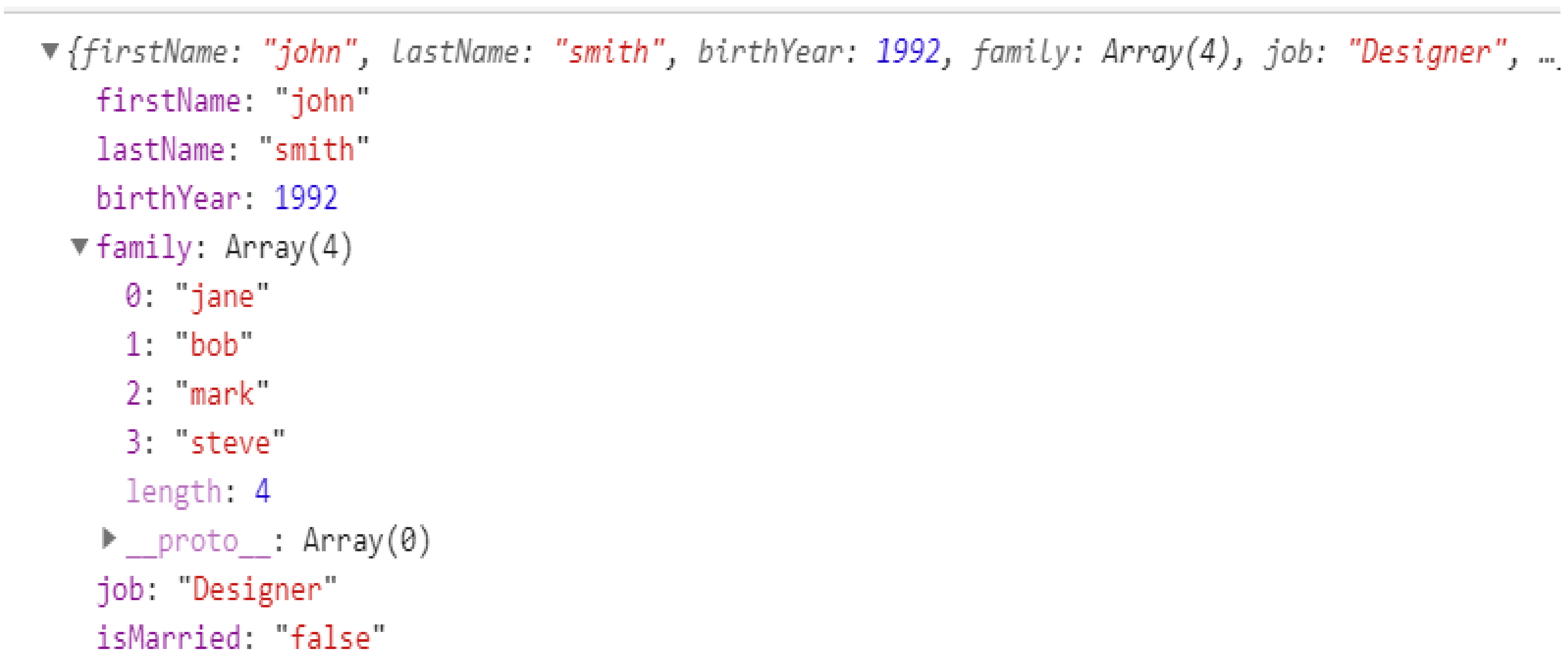
Objects & Properties

Object properties are defined as a simple association between name and value.Properties, refer to the collection of values, which are associated with the JavaScript object.



```
var john = {

     firstName : 'john',
     lastName : 'smith',
     birthYear :1992,
     family : [ 'jane','bob','mark','steve' ],
     job : 'Designer',
     isMarried : 'false'
 }


console.log(john);
```

output



```
▼{firstName: "john", lastName: "smith", birthYear: 1992, family: Array(4), job: "Designer", …
   firstName: "john"
   lastName: "smith"
   birthYear: 1992
  ▼family: Array(4)
     0: "jane"
     1: "bob"
     2: "mark"
     3: "steve"
     length: 4
   ▶__proto__: Array(0)
   job: "Designer"
   isMarried: "false"
```

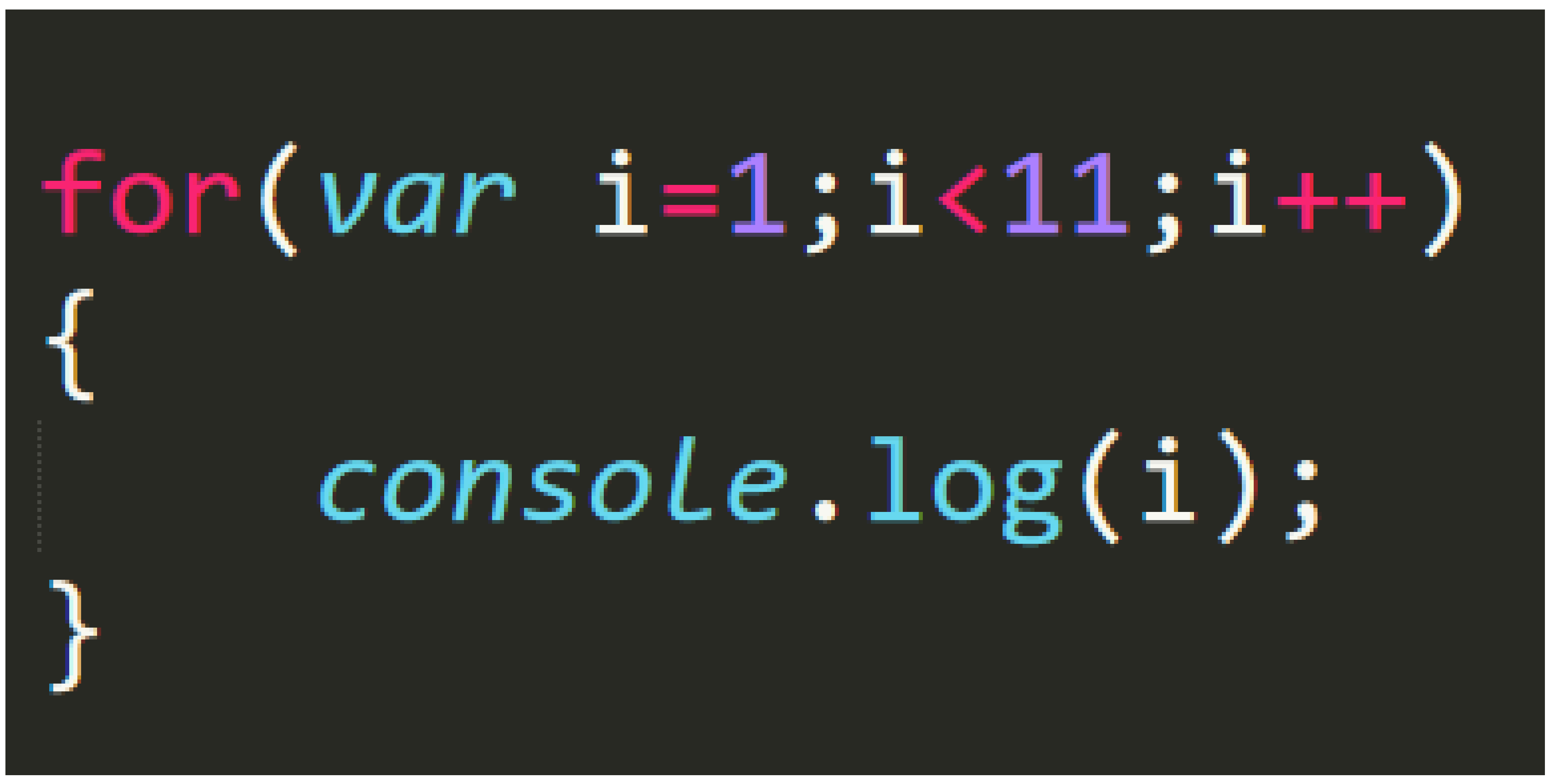
we can also console.log(john.isMarried) output will be false or console.log(john.firstName) output will be john so we can target separate elements we can change any value by simply :-
john.firstName = ‘ Mark ’;

rohit_virkud

Loops & iteration

Loops - Helps to automate repetitive tasks.

Display 1 to 10 numbers
now here we have to put console.log(1) ..then 2,3 and so one
this is a repetitive task lets code it with loops



```
for(var i=1;i<11;i++)
{
    console.log(i);
}
```

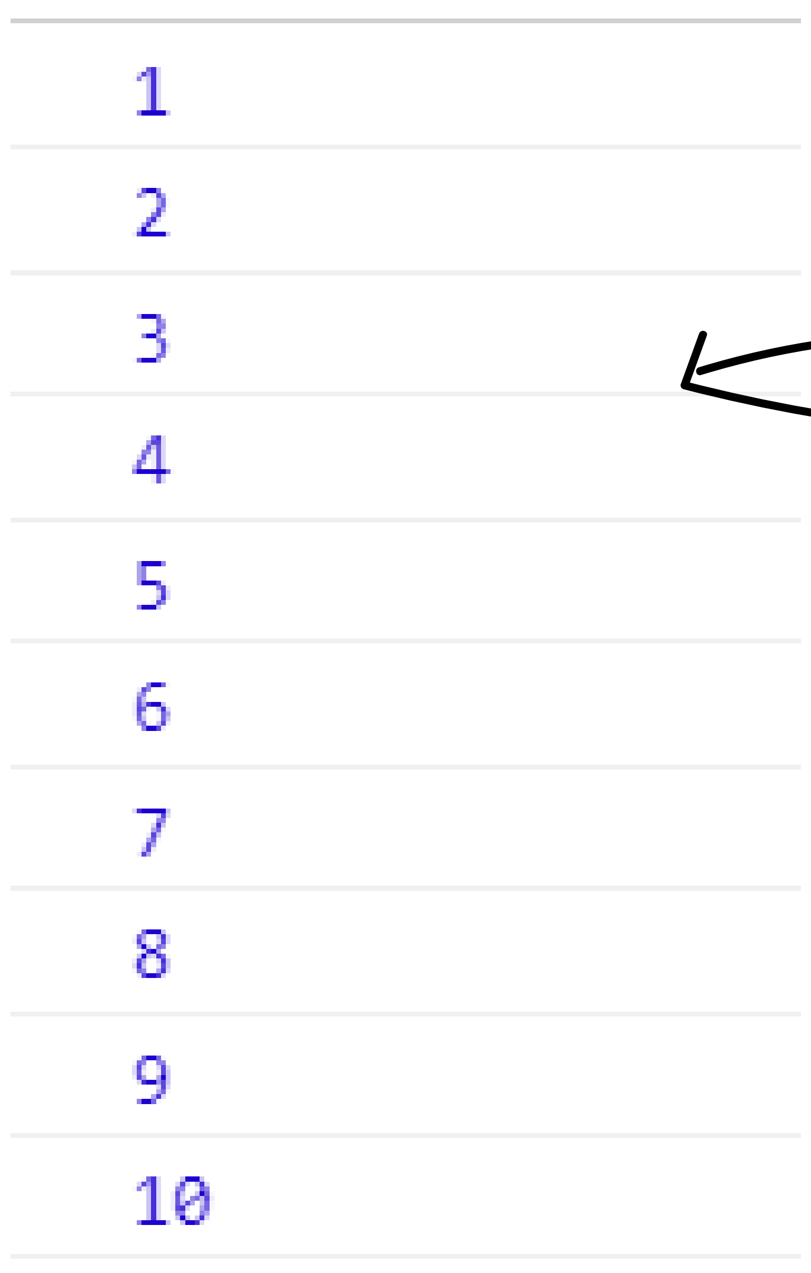
so basically here ,
lets break the code

i = 1 // declaration
i < 11 // condition so currently i = 1 so condition is true

i++ // i get incremented

```
{
   console.log(i) //prints the value 1
}
```

Here is the twist loop will not break until i becomes greater than 11 or condition becomes false, so now i = 2 and condition is again true i<11 then i++ means i get incremented by 1 so Now i = 3 and in { prints the value 2 }...and it will work until it prints all values below 11

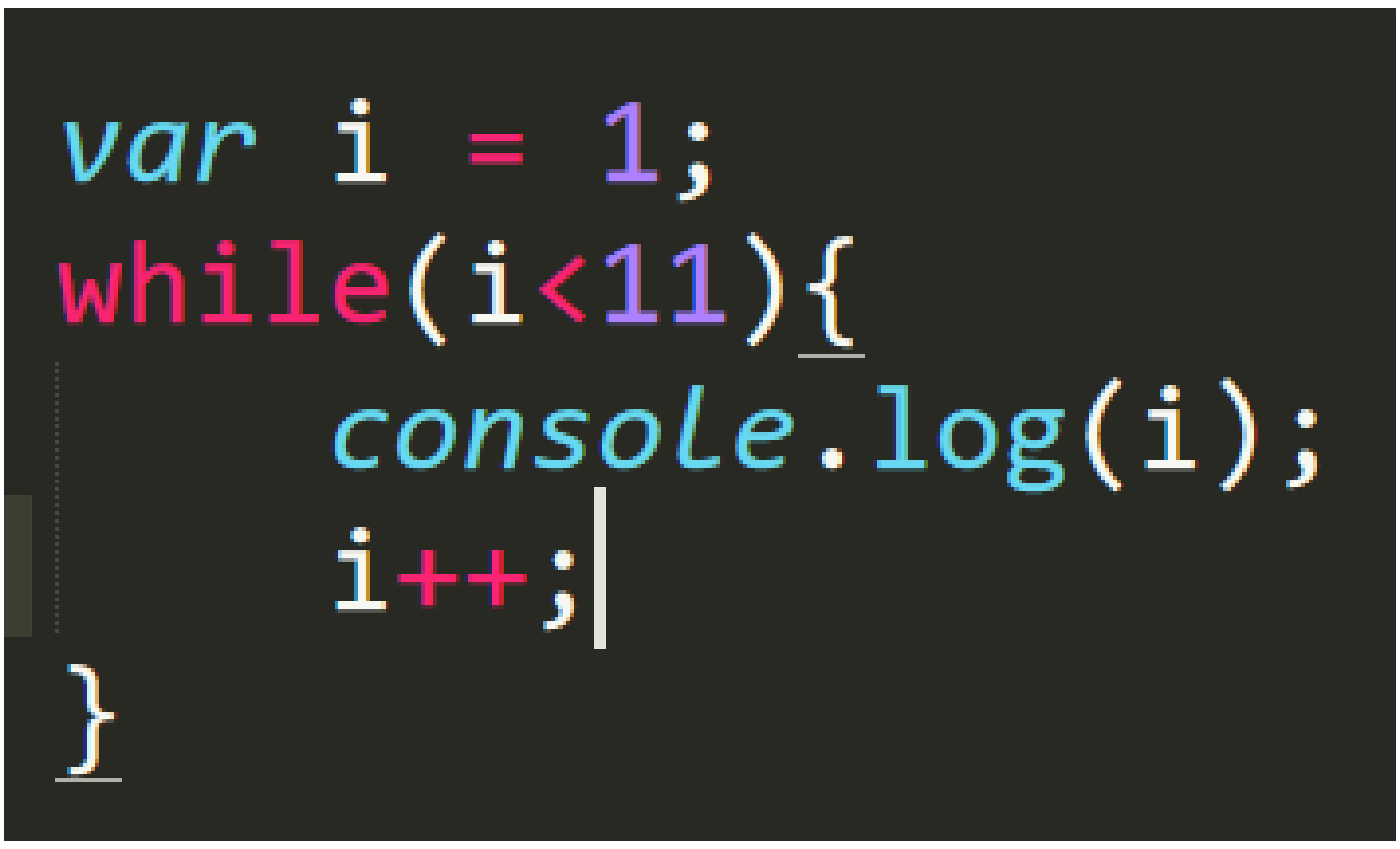


```
1
2
3
4
5
6
7
8
9
10
```

OUTPUT

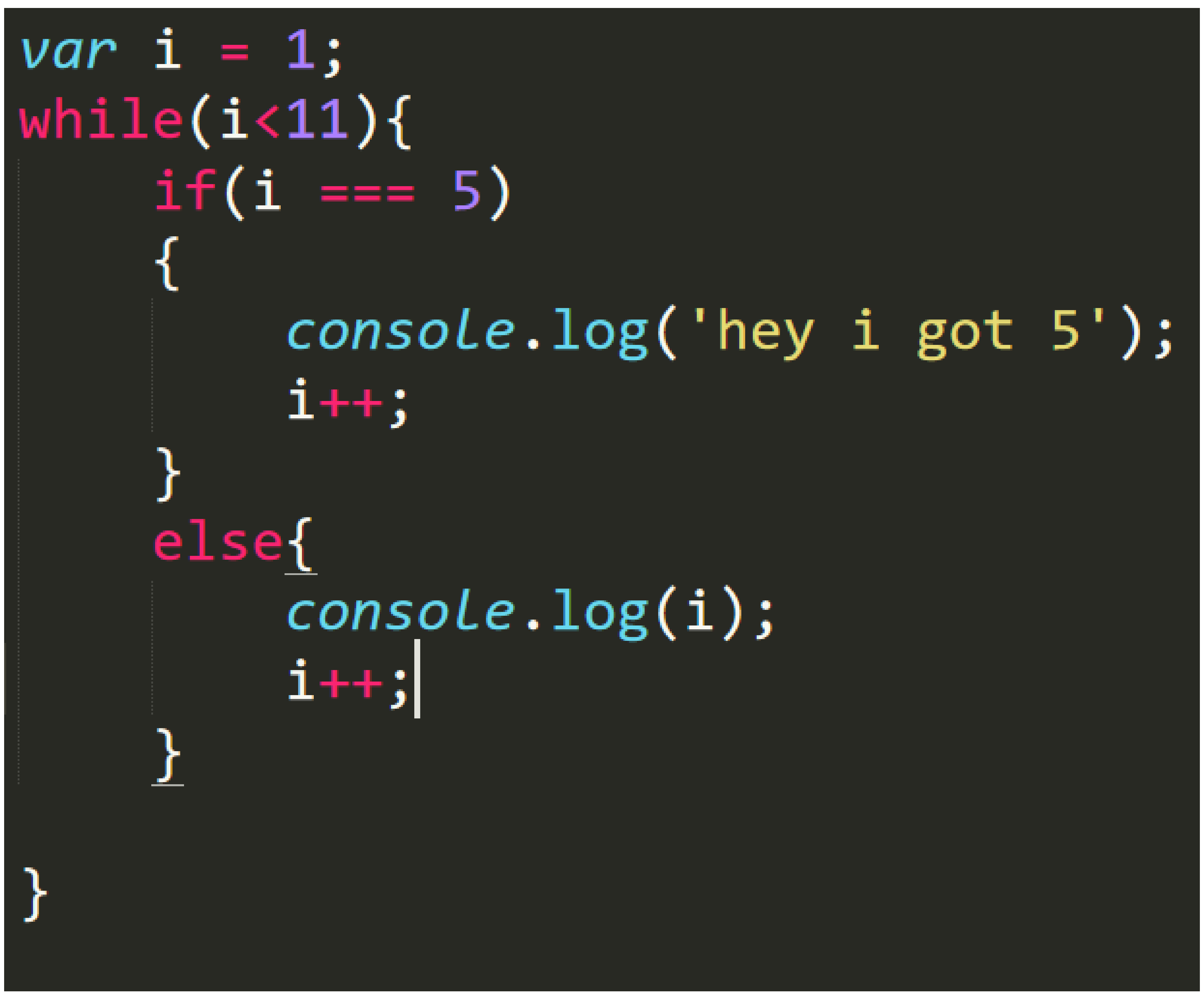
rohit_virkud

While loop



```
var i = 1;
while(i<11){
    console.log(i);
    i++;
}
```

while loop is a control flow statement same as for loop
just change in syntax a bit and works on given
Boolean codition



```
var i = 1;
while(i<11){
    if(i === 5)
    {
        console.log('hey i got 5');
        i++;
    }
    else{
        console.log(i);
        i++;
    }

}
```

we can also use conditions inside loops

rohit_virkud

Thanks for downloading this free javascript book

made with ❤ by Ro-HiT Virkud