# JS Objects

Object is all about key value pair

* In JS everything is an object so that if you understand the object, you will understand JavaScript
* Use object literals to create an object do not use new key word to create an object
* You cannot use two same key value pair in a { } object
* You can create property in object and access with it by following n

# Create and put key value pair or property

e.g.

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

  cars:{

    car1: "farari",

    car2: "Ford",

    car3: "Toyota"

  }

}

//ways To put values in object

//you can access object values

//also with these

myObj.subject = "English";

myObj["subject2"] = "Math";

myObj["cars"]["car4"] = "Honda"

console.log(myObj);

* You can put function in object by
* Function inside the object called object method

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

}

//put function in object

myObj.fun1 = function(){

  return "adil"

}

myObj["fun2"] = function(){

  return "younas"

}

console.log(myObj);

# Change key value pair

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

  today: function(){

    return "ali"

  }

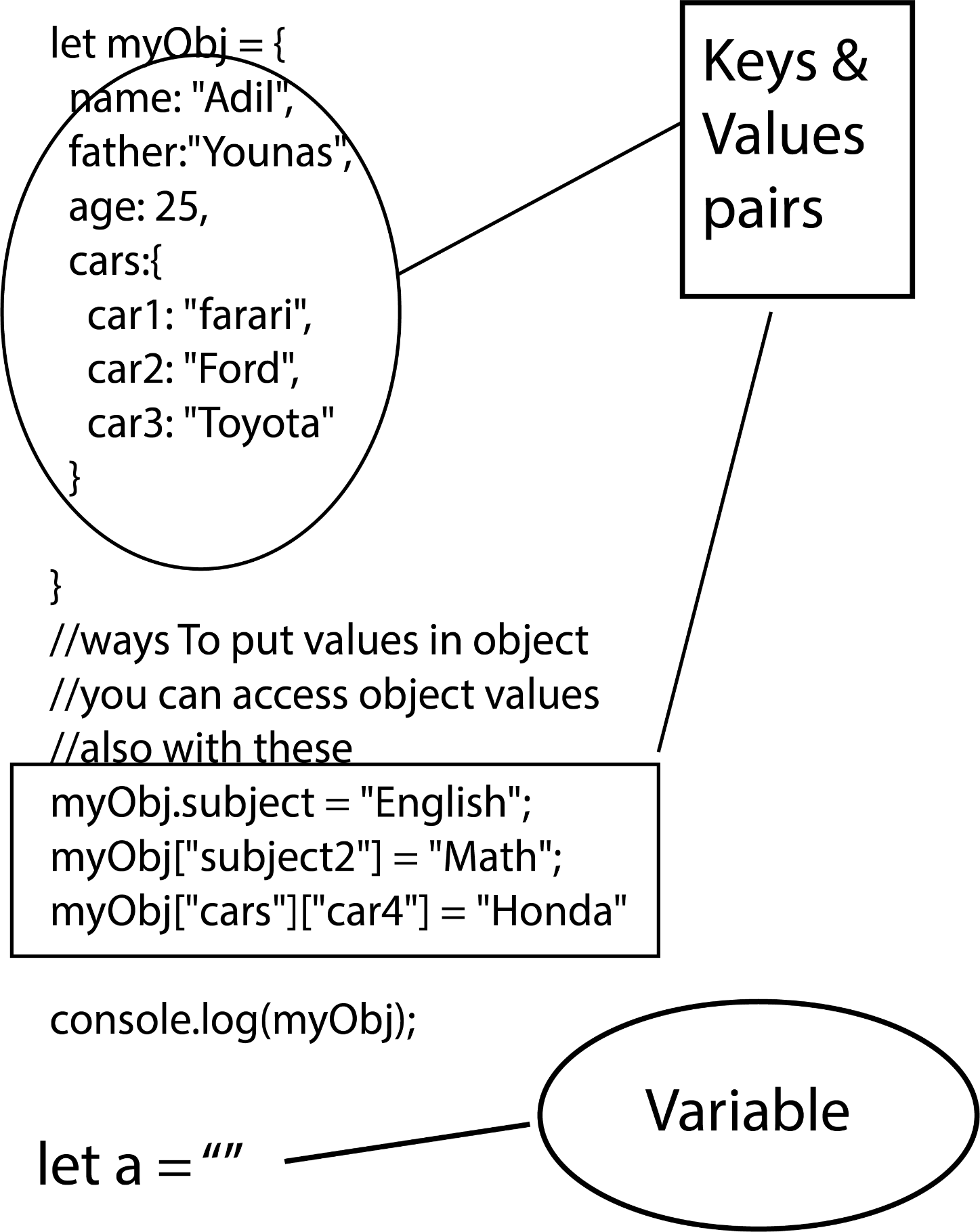
}

1. myObj.name = "haramoo"

2. myObj.today = myObj.today.toString()

console.log(myObj);

* Beware of variable and key value pair
* If you think to change or put value by storing key to variable then it is bad so that the following example will lead you to heaven path.



# Delete key value pair

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

}

//delete key value pair

delete myObj.age

console.log(myObj);

# Iteration of object using for in

* It is hard to iterate object using forEach(), for of or map it will give a type error:

const user = {

name: 'John Doe',

email: 'john.doe@example.com',

age: 25,

dob: '08/02/1989',

};

for(let x in user){

  console.log(x + " " + user[x]);

}

# Conversation of object

1. Object to Array by using

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

}

//object to array

let Array = Object.values(myObj)

console.log(Array);

//output =>['Adil', 'Younas', 25]

1. Array to object

    let obj ={

      name:"adil",

      class:"msc"

    }

    let b = Object.values(obj)

    console.log(b); //to array

    let x = Object.assign({},b)

    console.log(x); //to object

1. Stringify Object (key value convert into string)

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

}

//object to array

let string = JSON.stringify(myObj)

console.log(string);

//output => {"name":"Adil","father":"Younas","age":25}

1. Stringify Date (key value convert into string)

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

  today: new Date()

}

//object to array

let string = JSON.stringify(myObj)

console.log(string);

//output => {"name":"Adil","father":"Younas","age":25,"today":"2022-08-30T11:47:33.746Z"}

1. Stringify function (key value convert into string)

let myObj = {

  name: "Adil",

  father:"Younas",

  age: 25,

  today: function(){

    return "ali"

  }

}

//stringify function is complecated

//1. you convert function to string using toString()

//2. JSON.stringify()

myObj.today = myObj.today.toString()

let strFunction = JSON.stringify(myObj)

console.log(strFunction);

//output => {"name":"Adil","father":"Younas","age":25,"today":"function(){\n    return \"ali\"\n  }"}

1. Stringify Array (key value convert into string)

let myArr = ["adil",25,"younas"]

console.log(typeof myArr); //object

let strArray = JSON.stringify(myArr)

console.log(typeof strArray); //String

~~skip Accessor~~

# JavaScript Object Constructors

* For good practice name the constructor function with upper-case
* Call when object is created
* By object literals you can create a single object with fixed values but with JS object constructor you can use one time created object again and again with different values in object every time.\
* You can also put values through prototype in class and object constructor
* To edit via prototype use Student and for read or access obj.something
* By prototype you add variable of function
* Object is use only for entering values

Best example for prototype in object constructor

function Person(first, last, age, eye) {

  this.firstName = first;

  this.lastName = last;

  this.age = age;

  this.eyeColor = eye;

  console.log(this.nationality);

}

Person.prototype.nationality = "English";

const myFather = new Person("John", "Doe", 50, "blue");

function Student(name,classs,subject,age){

  this.stuName = name

  this.stuClass = classs

  this.stuSubject = subject

  this.stuAge = age

  console.log(name +" " +  classs + " " + subject);

}

let stu = new Student("Muhammad Adil","Master","English",55)

let stu2 = new Student("Muhammad Aqil","Inter","Math",25)

Warning

* It is not like class constructor and prototype
* It is only use to use object for multi-people or different data all time
* If you create an function inside object constructor then use this.info as a variable
* Oops use simple prototype and object constructor is like an object

function Student(name,classs,subject,age){

  this.stuName = name

  this.stuClass = classs

  this.stuSubject = subject

  this.stuAge = age

  console.log(name +" " +  classs + " " + subject);

  this.info = function info(){

    console.log(name);

  }

}

let stu = new Student("Muhammad Adil","Master","English",55)

stu.info() // stu.info() is not a function

# Put values and function in object constructor

* Above warning part distinguish object constructor from classes and put values in object constructor distinguish it from normal Object
* Student.name = “Hadia” & Student.fun = function () { } will not work so key prototype is use between them
* I am not putting a value I put a whole prototype in it which consist all the values of object itself

  function Person(first, last, age, eyecolor) {

  this.firstName = first;

  this.lastName = last;

  this.age = age;

  this.eyeColor = eyecolor;

}

//for key value

Person.prototype.nationality = "English";

//for function

Person.prototype.name = function() {

  return this.firstName + " " + this.lastName;

};

# Prototype & Inheritance

“Prototype”

* Prototype is like an object, array, function etc. itself + its hidden features
* Prototype often use in inheritance.

“Inheritance”

* If you want to use object inheritance then it is ok with this type
* if you use object constructor then inheritance is possible through prototype.
* What if you use object and inherit the value of array

// object inheritance

let obj = {

  name:"adil",

  class:"msc"

}

let obj2 = {

  subject:"MSC",

  \_\_proto\_\_:obj

}

console.log(obj2.name);

now

// this is inheritance using constructor method of object

let obj = {

  name:"adil",

  class:"msc"

}

function Person(first, last, age, eyecolor) {

  this.firstName = first;

  this.lastName = last;

  this.age = age;

  this.eyeColor = eyecolor;

}

Person.prototype = obj

let a = new Person("adil","younas",25,"black");

console.log(a);

console.log(a.name);

Now

//how object take array prototype

let arr = [2,4,"adil"]

function Person(first, last, age, eyecolor) {

  this.firstName = first;

  this.lastName = last;

  this.age = age;

  this.eyeColor = eyecolor;

}

Person.prototype = arr //object taking array prototype

let a = new Person("adil","younas",25,"black");

console.log(a);

console.log(a[2] + " " + "and its age is" + " " +   a.age);

~~Skip Iterables~~

# Set and Map

* Set method is use to store unique values from array and object it self can’t take same values if you grant it then it will give you filter object in console
* Array of object is not Acceptable
* Object in array somewhere is acceptable
* Create an set is easy

**How to create a Set**

* Passing an Array to new Set()
* Create a new Set and use add() to add values
* Create a new Set and use add() to add variable

const letters = new Set();

* To achieve uniqueness from array pass the array into it

let a = ["a","b","c"]

const letters = new Set();

* Add or access data of array or object

//to add something in object we use

//because obj.father bring change in object prototype somehow

let obj = {name:"adil"}

obj.father = "younas"

console.log(obj);

//to add something in array we use

//because array[3] bring change in object prototype somehow

let array = [1,2,3]

array[3] = 4

console.log(array);

* Add data into set() method is also different in array and object we manipulate with them but here after creating an object of set we manipulate with it

//to add array in set we use

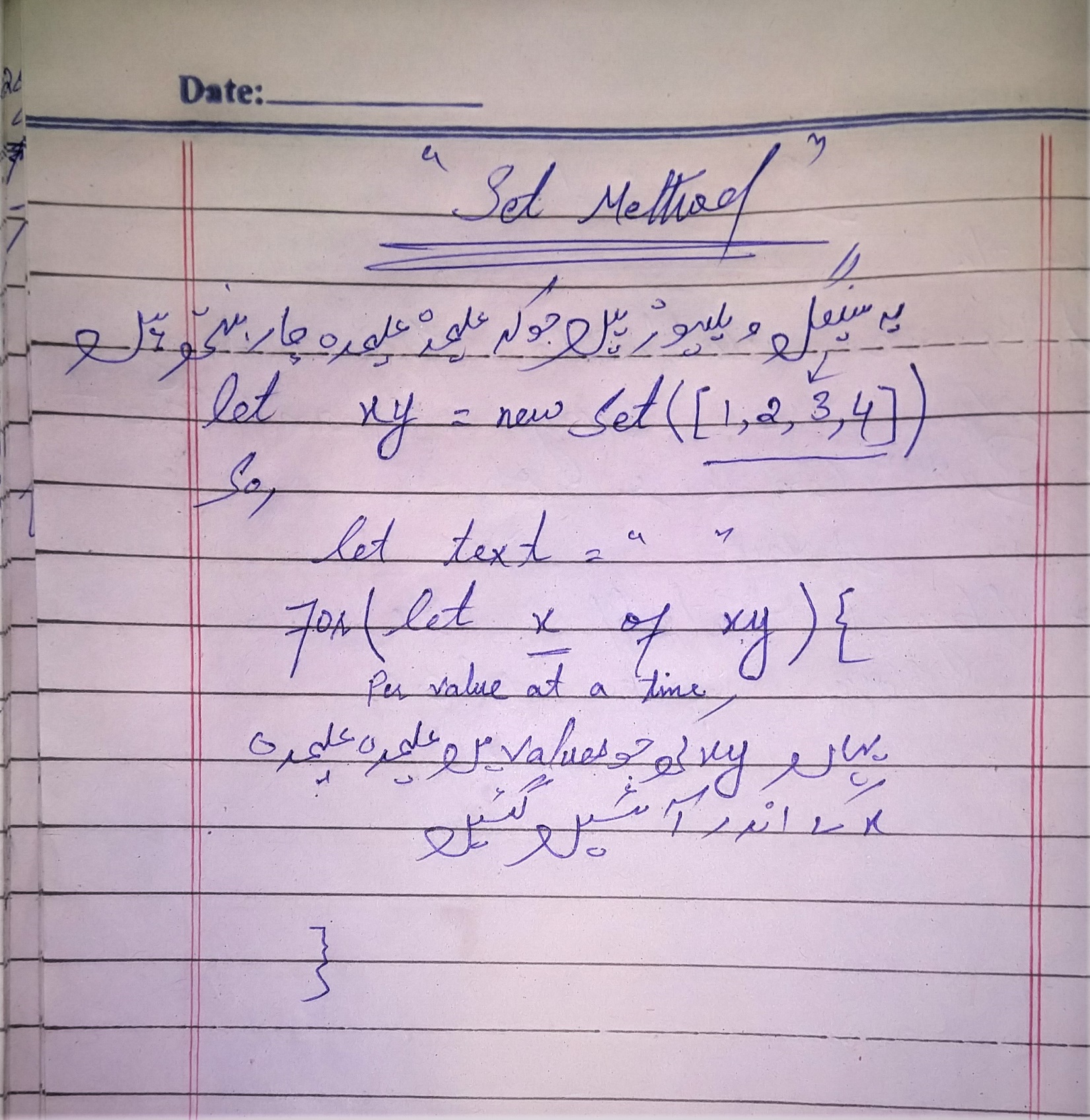
let array = [1,2,3]

let newset = new Set(array)

newset.add(45)

console.log(newset);

loop



Some methods use in Set()

1. Add() method add variable or element
2. Delete() method delete variable or element
3. Has() method return true if value exist
4. Clear() method clear set
5. Size property return number of elements in set

# Map()

**How to create a Map**

* Passing an Array to new Map()
* Create a new Set and use add() to add values
* Create a new Set and use add() to add variable is bad and show undefine so do not add variables
* Key value pair live in [ ]

//first way

const letters = new Map([

  ["a","b"],

  [40,45]

]);

console.log(letters);

//second way

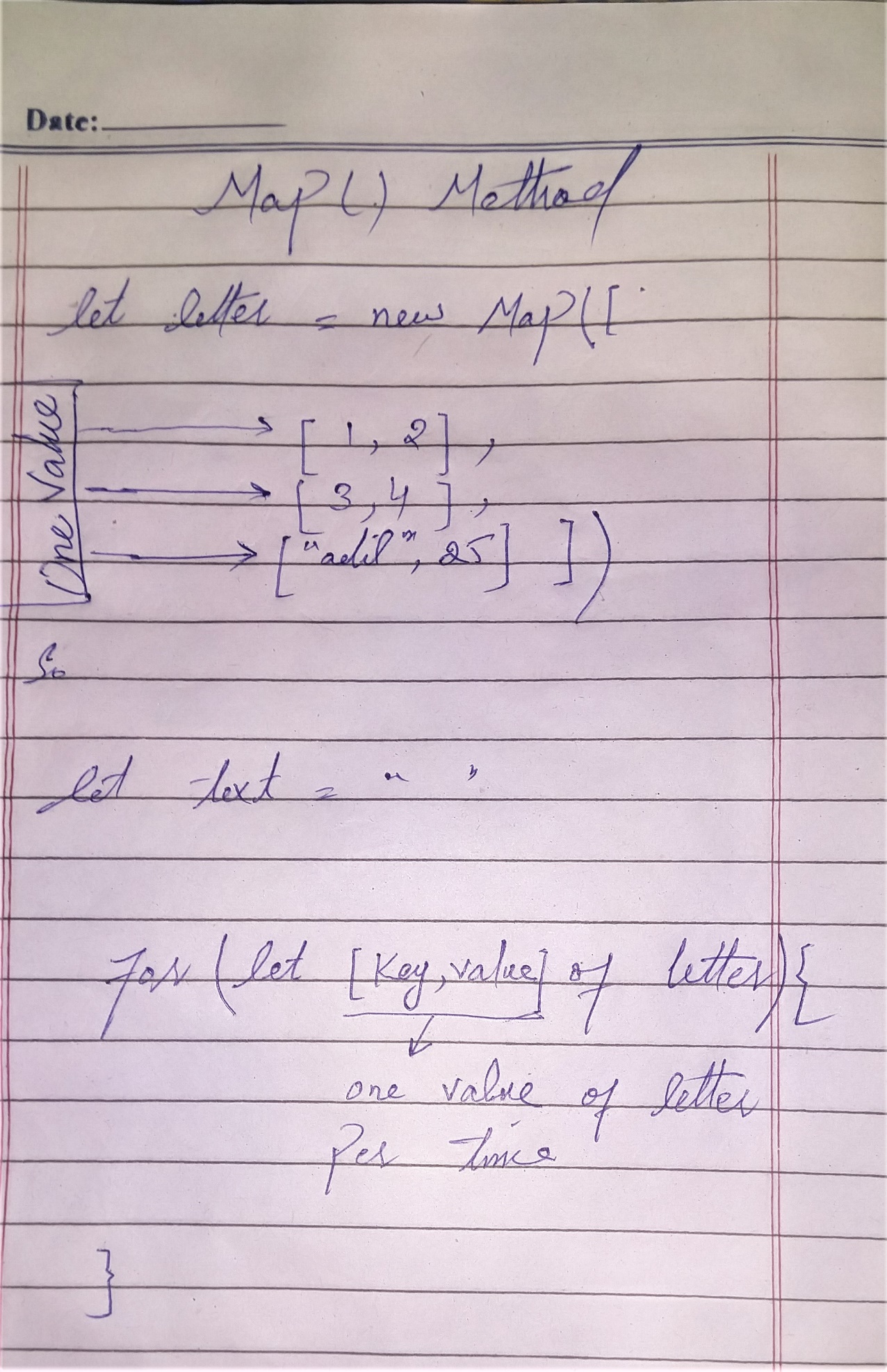
let variable = new Map();

variable.set("adil","msc");

variable.set("aqil","Intermediate");

console.log(variable);

Loop



const letters = new Map([

  ["a","b"],

  [40,45]

]);

let text = ""

for(let [key,value] of letters){

  text += key + "<br>" + value + "<br>"

}

document.write(text)

Some methods use in Map()

1. Set() method add element pair
2. Get() method get the value for you
3. forEach() iterate it
4. Delete() method delete variable or element
5. Has() method return true if value exist
6. Clear() method clear set
7. Size property return number of elements in set